University of Canterbury
Department of Mathematics and Statistics

# Topics in Lipschitz Global Optimisation

A Thesis Submitted for the Degree

of Doctor of Philosophy

at the University of Canterbury

1995

by

Baoping Zhang

Supervisor : Professor G.R.Wood

# Abstract

The global optimisation problem has received much attention in the past twenty-five years. The problem is to find an algorithm which will locate the overall, or global, optimum of a function. It can be viewed as a computer-age approach to the problem approached analytically by Newton over three hundred years ago. While Newton's algorithm deals with convex optimisation problem, the development of global optimisation approaches cover more general cases. It is a problem of considerable practical importance – chemists, physicists, engineers and statisticians need answers to such minimisation or maximisation problems.

The problem is insoluble without imposing some regularity conditions on the function to be optimised. The simplest such regularity condition is that of Lipschitz continuity – an acknowledgement that the function cannot vary by more than a fixed amount as the independent variables move in a fixed region.

The most acknowledged pioneer work on Lipschitzian global optimisation for objective functions of one variable was independently produced by Piyavskii and Shubert in 1967 and in 1972 respectively. The former Soviets have placed great emphasis on this approach to the problem, relating many of their methods back to the work of Piyavskii. A number of higher dimensional generalisations of the Piyavskii-Shubert algorithm have since been developed, including those due to Mladineo (1986), Pintér (1986) and Wood (1992).

Apart from deterministic approaches to the global optimisation problem for Lipschitz continuous functions mentioned above, active research is being conducted to find stochastic methods to solve this problem. The pure adaptive search analysed by Zabinsky and Smith offered considerable hope for researchers pursuing practical stochastic methods to solve the problem. Pure adaptive search gives a linear

complexity convergence result in dimension. For deterministic algorithms the computational complexity of convergence is exponential.

A general introduction to the field of global optimisation and to the more specific topics studied in this thesis is presented in the first chapter. Chapters two and three are devoted to the development and properties of Lipschitz based algorithms. A survey of selected deterministic and stochastic Lipschitz global optimisation algorithms is first presented. The tightest lower bounding function of an objective function after a set of evaluations and a necessary condition for finite convergence of Lipschitz based approaches is obtained.

The multidimensional bisection algorithm of Wood is discussed in detail in Chapter four. The context and acceleration strategies for the multidimensional bisection algorithm are presented. The performance of numerical acceleration methods is described for certain test functions. The context of multidimensional bisection is studied by showing such algorithms fall in a broadened branch and bound framework. A higher dimensional localisation convergence result is obtained.

A stochastic method, pure localisation search, is presented and studied in Chapter five. This is an attempt to find an efficiently implementable algorithm, based on a stochastic variant of the Piyavskii-Shubert algorithm, which can realise the desirable linear complexity in dimension of pure adaptive search. Comparison of pure localisation search with pure adaptive search, pure random search and the Piyavskii-Shubert algorithm is given.

Estimation of the Lipschitz constant of a function is investigated in Chapter six. It is itself a global optimisation problem. Existing methods for such estimation are surveyed. A new stochastic approach is developed. The distribution of the largest absolute slope in a fixed size sample of absolute slopes is shown to approximate a well known distribution. The location parameter of this distribution is used

as an estimate of the Lipschitz constant. The applicability of this method is studied for univariate and multivariate functions. Numerical results are presented.

# Statement of originality

Some of the contents of this thesis are jointly researched with other authors. Chapter 4 relates joint work with G.R. Wood and W.P. Baritompa. My contributions to this are the the numerical implementation, the idea of extending the Horst and Tuy branch and bound framework to the language of covers and the proof of the localisation properties of multidimensional bisection. Chapter 5 involves joint research with W.P. Baritompa, R.H. Mladineo, G.R. Wood and Z.B. Zabinsky. My contributions to this are parts of the proof of Theorem 5.3.1, the implementation of the comparison of the PRS, PAS, PLS and PS algorithms for the test functions and the implementation of PLS for the witch's hat. Chapter 6 is joint work with my supervisor. This work is largely carried out by myself. Unless mentioned above, the material in this thesis is the product of my own scholarship and research.

# Acknowledgement

I would like to express my sincerest thanks to Professor G.R. Wood, for his assistance and supervision over the years taken for this research. His advice and guidance, together with his helpful criticisms, proved most valuable and are much appreciated.

I would also like to thank Dr. P. Renaud and Professor R. Kerr for their encouragement and support of my study. Special thanks are due to Dr. W.P. Baritompa in the Department of Mathematics and Statistics for the later supervision of my thesis and helpful discussions during the joint work.

The award of a Ministry of External Relations and Trade fees scholarship during my study enabled me to carry on my postgraduate study in New Zealand. I gratefully acknowledge that.

Finally, thanks to my wife and my parents for their encouragement and support.

# Contents

CHAPTER 1

# Introduction

Global optimisation involves finding the overall minimum (or maximum) value of a multivariable real-valued objective function over a constraint set which is generally assumed to be compact. Many practical problems can be posed as global optimisation problems. For example, the composite laminate design problem discussed in [83] in the field of engineering design problems, the maximisation of yield in agriculture, where yield is viewed as a function of the amount of fertilizer used [11] and the minimum potential energy conformation of a molecule in chemistry [39, 67]. The search to find an efficient algorithm along with a computer code has led to more and more research activity in this area in recent decades. Global optimisation algorithms are different from local optimisation approaches. Local optimisation algorithms usually reach a local minimum, local maximum or saddle point, depending on the starting point. A successful global optimisation algorithm, however, should possess a criterion or procedure which can ascertain whether global extremes of the objective function are obtained.

A number of excellent survey works on global optimisation have appeared in recent years. These include [16], [20], [33], [35] and [63].

It is well known that global optimisation problems belong to the class of NP-hard problems, see for instance [63]. This means that the computation time required

to solve the problem grows exponentially as the the size of the input data increases. This inherent difficulty forces global optimisation researchers to add assumptions upon the objective function and the domain.

One commonly used assumption is Lipschitz continuity of the objective function, as it is considered moderate and acceptable from the point of view of the real world. It requires the slope of the function determined by every pair of points in the domain $D$ to be uniformly bounded. Throughout this thesis we deal with global optimisation problems based on this assumption, and discuss and develop both deterministic and stochastic approaches for solving such problems.

### The basic problem

Let $f$ denote a real-valued function, defined over a compact convex subset $D$ of $\mathbf{R}^d$, satisfying the Lipschitz condition

$$|f(x_2) - f(x_1)| \leq M||x_2 - x_1|| \quad \text{for all} \quad x_1, x_2 \in D,$$

where $M$ is a positive constant, and $||\ ||$ is the Euclidean norm. We denote the collection of such functions with Lipschitz constant $M$ by $L(M)$. Notice that for any $M' > M$, if $f \in L(M)$ over $D$ then $f \in L(M')$. We term $\inf\{M : f \in L(M)\}$ over $D$ the least Lipschitz constant of $f$ over $D$. We are interested in the global minimisation of $f$ over $D$. We assume that $f$ can be evaluated on $D$ but that an explicit analytical expression for $f$ may not be available.

### Classes of problems

There are two classes of problem, related to the minimisation of a Lipschitz function, considered in the literature. The first involves finding the optimal function value $f_*$ together with an optimiser $x_* \in D$. The second involves finding the localisation of the optimiser(s), the set $E = \{x_* \in D \ : \ f(x_*) = f_*\}$. Each problem has two

versions: the determination of exact solutions and the determination of approximate solutions. We denote them using the notation Hansen, Jaumard and Lu introduced in [29].

**First class of problems:**

Problem $P$. Find $f_*$ and an $x_* \in D$ such that

$$f(x_*) = f_* = \min_{x \in D} f(x).$$

Problem $P'$. Given $\epsilon > 0$ find $f_{opt}$ and an $x_{opt} \in D$ such that

$$f_{opt} = f(x_{opt}) \leq f_* + \epsilon.$$

An algorithm will be said to be $\epsilon$-*convergent* if it solves problem $P'$ in a finite number of iterations.

**Second class of problems:**

Problem $Q$. Find $E$.

Problem $Q'$. Find subsets $D_i$ of $D$, for $i \in I$, such that $E \subseteq \bigcup_{i \in I} D_i$ and $\mu(\bigcup_{i \in I} D_i) - \mu(E) \leq \eta$, where $\eta$ is a small positive number and $\mu$ is Lebesgue measure.

An algorithm will be said to be $\eta$-*convergent* if it solves problem $Q'$ in a finite number of iterations.

Note that finding an exact solution in finitely many steps (finite convergence) is mainly of theoretical interest.

## Algorithms

Algorithms for the solution of global optimisation problems can be divided into two classes in philosophy: deterministic and stochastic. A deterministic algorithm uniquely prescribes the next evaluation point and does not involve any stochastic concepts. Most deterministic algorithms provide for convergence or $\epsilon$-convergence,

under a certain termination criterion. That is, the global optimum value of the objective function can be obtained or a value can be found within a certain distance of the global optimum value. Stochastic approaches take the next evaluation point(s) randomly drawn from a certain distribution; thus they usually need fewer assumptions about the objective function but sacrifice an absolute guarantee of success. Under mild conditions on the distribution and the objective function $f$, however, the probability of convergence to a close approximation of the global minimum is proved to be one. For instance, see [72]. Related works can also be found in [4], [9], [14], [15], [40], [51] and [53]. A broad review of this field is provided in [87].

From the viewpoint of methodology, algorithms for solving global optimisation problems can also be divided into two categories: passive and sequential. A passive algorithm uses a fixed procedure to evaluate the objective function. Such a procedure determines the next evaluation point(s) and does not use the information obtained before the current evaluation. It has the advantage that it is simple to describe and does not involve too much manipulation at each step, but has the disadvantage that it does not use available information which may hasten convergence. A sequential algorithm uses information from the evaluation points and the function values available up until the current step.

## Outline of this thesis

In Chapter 2, a survey of certain deterministic and stochastic Lipschitz global optimisation algorithms will be presented. Horst and Tuy's branch and bound framework is also reviewed in this chapter; it is a preliminary for proving that the Wood algorithm lies within a broadened such branch and bound framework in Chapter 4.

We discuss some convergence properties of Lipschitz based algorithms in Chapter 3. The tightest lower bounding function is presented for an objective func-

tion of several variables and necessary conditions for a sequential algorithm to solve problem $P$ or $Q$ in a finite number of steps is obtained. These generalise the results obtained in [29] from the univariate case to the case of higher dimensions.

As a special deterministic Lipschitz based approach, Wood's multidimensional bisection algorithm is studied in detail in Chapter 4. Multidimensional bisection can be viewed as a generalisation of the familiar "interval-halving" bisection method to higher dimensions [79]. Acceleration strategies and the effectiveness of the algorithm are investigated. Numerical results are presented for a set of test functions. The link between multidimensional bisection and other algorithms in the literature is discussed. We extend Horst and Tuy's framework to a "covering" format and show that multidimensional bisection then falls into this framework. A localisation result of Basso [8] is extended which ensures convergent localisations in higher dimensions.

A stochastic method, pure localisation search (PLS), is studied in Chapter 5. This is an attempt to produce a practical implementation of a pure adaptive search (PAS) styled algorithm [81]. Pure localisation search chooses the next evaluation point uniformly from the localisation $L_k$, the subset of $D$ which is known to still contain the global minima, using the provided information. Pure adaptive search occurs when we are always able to choose the next evaluation point according to a uniform distribution on the "improving set" within $D$. An improving set is the subset of the domain $D$ where the function value is strictly less than the least function value to date. It was shown in [81] that when pure adaptive search is applied to global mathematical programs satisfying the Lipschitz condition, the expected number of iterations to convergence increases at most linearly in the dimension of the problem, a desirable complexity result. The motivation to introduce pure localisation search is an attempt to find an efficiently implementable algorithm,

based on a stochastic variant of the Piyavskii-Shubert algorithm, which can realise the desirable complexity of PAS. We compare the performance of PAS, PLS, PRS and the Piyavskii-Shubert algorithm for a set of test functions.

For every Lipschitz based algorithm, the Lipschitz constant is assumed known. In practice, this is not often the case. In fact it is known that to find the best Lipschitz constant of a function itself is a global optimisation problem. This statement will be discussed in Chapter 6. It is critical to have an estimate of the Lipschitz constant of a function for every Lipschitzian algorithm. It is also necessary to have a good estimate, since the better the estimate the faster the algorithm converges. In Chapter 6, existing methods dealing with Lipschitz constant estimation problems are outlined and classified. A stochastic method for estimating the Lipschitz constant based on the theory of extreme value distributions is presented. We show that the largest absolute slope in a fixed size sample of slopes has an approximate Reverse Weibull distribution. Such a distribution is fitted to the largest slopes and the location parameter used as an estimate of the Lipschitz constant. The applicability of this approach is discussed. For the univariate case, a wide class of functions is shown to satisfy the necessary condition on the objective function. For the multivariate case, a partial result is obtained for linear functions. Numerical results are presented.

In the last chapter, we summarise the contents of this thesis.

CHAPTER 2

# A survey of selected Lipschitz global optimisation methods

Research on Lipschitz based global optimisation is diverse. A survey of Lipschitz based algorithms for solving one dimensional global optimisation problems is provided in [31]. A comprehensive survey on the methods designed for both univariate and multivariate Lipschitz optimisation is discussed in [28], by Hansen and Jaumard. This forms a chapter on Lipschitz optimisation in [33].

Deterministic approaches to problems $P$ and $P'$ defined in Chapter 1 have been developed by Evtushenko [19], Piyavskii [61], Pintér [52], Shubert [69], Timonov [76], Shepilov [70], Strongin [75], Danilin [13] and Hansen, Jaumard and Lu [32] for the univariate case. In addition Hansen, Jaumard and Lu [29] studied the number of iterations of Piyavskii-Shubert's algorithm [29]. Basso [8], Galperin [22, 23], Shen and Zhu [68] considered problems $Q$ and/or $Q'$ defined in Chapter 1. Some of the above algorithms have been extended to the multidimensional case. For example, Galperin [22], Pintér [55, 56, 60], Mladineo [43], Wood [78, 79], Meewella and Mayne [42], Mayne and Polak [41], Shepilov [70] and Strigul [73].

For stochastic approaches to Lipschitz global optimisation, Mladineo in [46] has proposed a "randomized cone algorithm', which initiates a study combining the Lipschitz condition with existing stochastic methods. Stochastic approaches

for solving general global optimisation problems usually require fewer assumptions about the objective function, so such algorithms can be applied to Lipschitz global optimisation problems $P$, $P', Q$ and $Q'$. Some examples of stochastic algorithms are pure random search [1, 11], the random sampling framework of Archetti, Betró and Steffé [2] and the estimating method using order statistics of de Haan [18]. An attractive approach is the pure adaptive search proposed by Zabinsky and Smith [81].

The aim of this chapter is to lay out the basic deterministic and stochastic Lipschitz based global optimisation methods upon which the investigation of this thesis rests. In Section 2.1, we recall some basic facts for Lipschitz global optimisation. In Section 2.2, we review a set of selected deterministic methods. The Horst and Tuy branch and bound framework is reviewed in Section 2.3. In Section 2.4, we discuss a set of selected stochastic methods.

## 2.1 Facts about Lipschitz global optimisation

For any objective function $f \in L(M)$, algorithms for solving Lipschitzian global optimisation problems use variations on two simple facts. Let $\mathcal{O} = \{(x, y) : y \geq M \parallel x \parallel, \text{ for } x \in \mathbf{R}^d \text{ and } y \in \mathbf{R}\} \subset \mathbf{R}^{d+1}$ be the cone at the origin with axis of symmetry the $y$-axis and spherical cross-section of radius one at height $M$, and suppose that $(x, y)$ lies on the graph of $f$ over $D$. Then

(1) no point inside $(x, y) - \mathcal{O}$ lies on the graph of $f$, and

(2) no point above $(x, y)$ can be a global minimum of $f$.

After evaluations at $x_1, x_2, \ldots, x_k$, with associated function values $f(x_1), f(x_2), \ldots, f(x_k)$, we can use these two facts to eliminate regions where we know that the global minimum of $f$ cannot lie. The region left after the elimination we term the

*bracket*, $U_k$. Figure 2.1 shows the case for dimension $d = 1$, where the shaded region is that eliminated and the blank region is the bracket $U_k$.

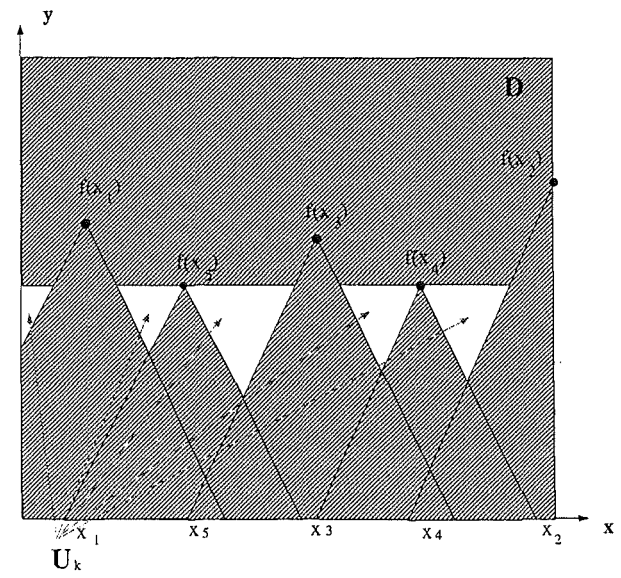



**Figure 2.1** The bracket $U_k$, after evaluations at $x_1, x_2, \ldots, x_k$, for the case $d = 1$ with Lipschitz constant $M = 2$.

The surfaces which form the lower envelope of bracket $U_k$, of $f$ (piecewise line segments when $d = 1$) lies on the function

$$F_k(x) = \max_{1 \le i \le k} \{f(x_i) - M||x - x_i||\}.$$

For a Lipschitz based algorithm, we let $L_k$ be the localisation after the $k$th evaluation. That is, $L_k$ is the projection of $U_k$ on $\mathbf{R}^d$. It is the smallest region at this point known to contain the solution set $E$. Let $L_\infty = \cup_{k=0}^{\infty} L_k$ and $A$ be the set of all accumulation points of the sample sequence.

## 2.2 Deterministic methods

We first review certain deterministic approaches to Lipschitz based global optimisation methods. The Piyavskii-Shubert algorithm is described at the outset, without

doubt a fundamental deterministic Lipschitz based algorithm. The Pintér algorithm, an axiomatic framework, is reviewed in this section. Two higher dimensional extensions of the Piyavskii-Shubert algorithm are highlighted, namely the Mladineo algorithm, a sphere-based extension, and the Wood algorithm, a simplex-based extension.

### Piyavskii-Shubert algorithm

Independently, Piyavskii [61] and Shubert [69] proposed the same algorithm for solving $P$ and $P'$ for univariate problems. They both use a piecewise linear lower bounding function. It is easy to determine the minimum of this lower bounding function. This gives the next evaluation point.

Given $f \in L(M)$ on $[a, b]$, consider problem $P'$ : find $f_{opt}$ and $x_{opt} \in D$ such that

$$f_{opt} = f(x_{opt}) \leq \min_{x \in D} f(x) + \epsilon.$$

**Initialisation:**

Evaluate at $x_0 = (a + b)/2$, let $f_*^0 = f(x_0)$, and $x_*^0 = x_0$.

Construct a lower bounding function $F_0$ given by $F_0(x) = f(x_0) - M|x - x_0|$, a piecewise linear function. Set $k = 1$.

**Iterations:**

At the beginning of step $k$, we have a set of evaluation points $x_0, x_1, \ldots, x_{k-1}$, corresponding function values $f(x_0), f(x_1), \ldots, f(x_{k-1})$ and a lower bounding function $F_{k-1}(x)$. Let the $k$th evaluation point be:

$$x_k = \arg\min_{x \in D} F_{k-1}(x).$$

If the location $x_k$ is not unique, choose one arbitrarily. Let $f_*^k = \min_{0 \leq i \leq k}\{f(x_i)\}$, the least function value to date, $x_*^k = \text{argmin}_{0 \leq i \leq k}\{f(x_i)\}$, a location of $f_*^k$ (if $x_*^k$ is

not unique, choose one arbitrarily) and $F_k(x) = \max_{0\le i\le k}\{f(x_i) - M|x - x_i|\}$, the lower bounding function, piecewise linear, at step $k$.

**Stopping Criterion:**

Let $F_*^k = \min_x F_k(x)$. For a pregiven tolerance $\epsilon > 0$, if the variation $V_k$ defined by $V_k = f_*^k - F_*^k$ is less than or equal to $\epsilon$, then stop. Otherwise increment $k$ and go on to the next iteration.

**Convergence Results:**

The algorithm guarantees:

$$1) \qquad \lim_{k\to\infty} f(x_k) = f_*, \quad \text{and}$$

$$2) \qquad \lim_{k\to\infty} \inf_{x\in E} |x - x_k| = 0$$

where $E$ is the set of minimisers of $f$ defined in Chapter 1. The convergence result is proved in [69].

This algorithm with related terms is illustrated in Figure 2.2 .

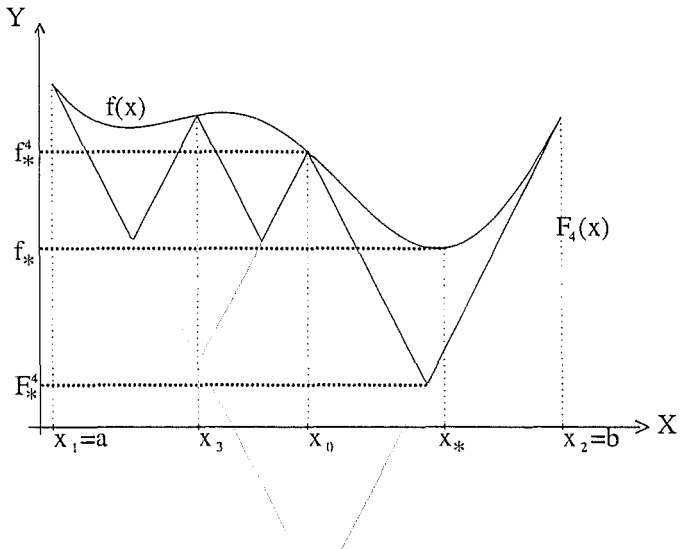



**Figure 2.2** Illustration of the Piyavskii-Shubert algorithm for a Lipschitz continuous function $f$ on $[a, b]$ after four evaluations at $x_0, x_1, x_2$ and $x_3$ with Lipschitz constant $M = 2$.

The Piyavskii-Shubert algorithm is a sequential algorithm. It is also a *one-step optimal* algorithm. The algorithm minimises, over all sequential algorithms, the

maximum of the variation $V_k$ over all functions which belong to $L(M)$ and passing through all evaluation points to date. The proof of the one-step optimality of the Piyavskii-Shubert algorithm is a special case of an optimality property proved by Wood in general dimension in [78, pp.171-172].

The Piyavskii-Shubert algorithm has the virtue that the lower bounding function $F_k$ is the tightest lower bounding function of $f$. That is, after $k$ evaluations, $F_k$ is the best lower bounding function, provided that the least Lipschitz constant of $f$ is used. This algorithm can be viewed as a branch and bound algorithm, as stressed by Horst and Tuy [36, pp.266-270]. At step $k$, the upper bound for $f$ on the current localisation is determined by $f_*^k$ and the lower bound by $F_*^k$. This forms the basis of the bounding strategy in the Horst and Tuy branch and bound framework. After each iteration, the Piyavskii-Shubert algorithm divides the problem into several similar subproblems. Subproblems with a lower bound greater than $f_*^k$ may be eliminated. Elimination provides the basis of the branching strategy in the Horst and Tuy branch and bound framework.

Research on the relationship between the number of function evaluations of the Piyavskii - Shubert algorithm and the number of function evaluations, $n_b$, of a best possible algorithm was initiated by Danilin [13]. A best possible algorithm is defined as an algorithm which takes the minimum number of evaluations, $n_b$, to guarantee an $\epsilon$-optimal value of an objective function $f$ has been reached. The number $n_b$ is used as a yardstick to evaluate the efficiency of sequential algorithms. For a given tolerance $\epsilon$, denote by $n'_{py}$ the number of function evaluations required to obtain a lower bounding function with minimum no less than $f_* - \epsilon$, and denote by $n_{py}$ the number of function evaluations required to meet the stopping criterion of the Piyavskii-Shubert algorithm, where $f_*$ is the unknown optimal value for problem $P'$. Danilin showed that $n'_{py} \leq 3n_b$. This inequality was improved to $n'_{py} \leq 2n_b + 1$

by Hansen, Jaumard and Lu in [31]. They also showed that $n_{py} \leq 4n_b + 1$. The sharpness of the above bounds was also proved by the same authors.

The Piyavskii-Shubert algorithm takes more iterations than a passive algorithm for a constant objective function. Such a passive algorithm evaluates the function $f$ at regularly spaced points $a + M/\epsilon, a + 3M/\epsilon, a + 5M/\epsilon, \ldots$. This result is described in [29].

### Pintér algorithm

Pintér in [52] proposed an axiomatically-described approach for solving problems $P$ and $Q$. The approach involves an interval-characteristic function $R$ and a point selection function $S$. The assumptions made about the interval-characteristic function and the point selection function make this class of algorithms convergent.

Given $f \in L(M)$ on $[a, b]$, consider problems $P$ and $Q$.

**Initialisation:**

Let $x^{(0)} = a$ and $x^{(1)} = b$. Evaluate $f(x^{(0)})$ and $f(x^{(1)})$.

Define a function $R$: $\mathbf{R}^4 \longrightarrow \mathbf{R}$, an interval-characteristic function. Define a function $S$: $\mathbf{R}^4 \longrightarrow \mathbf{R}$, a point selection function. Set $k = 1$.

The purpose of introducing $R$ and $S$ is to set a rule for selecting evaluation points to carry on an algorithm. An interval-characteristic function $R$ maps two updated adjacent evaluation points and their objective function values to a real number which provides a characteristic number to the interval formed by these two evaluation points. A point selection function $S$ maps each two updated adjacent evaluation points and their objective function values to a point in $[a, b]$ which is a candidate for the next evaluation point.

For example, the interval-characteristic function $R$ can be chosen as

$$R(s_1, s_2, y_1, y_2) = -\frac{1}{2}[M(s_1 - s_2) + y_1 + y_2]$$

and the point selection function $S$ can be chosen as

$$S(s_1, s_2, y_1, y_2) = \frac{1}{2M}[M(s_1 + s_2) + y_1 - y_2].$$

**Iterations:**

After the $k$th evaluation at $x^{(k)}$, rearrange $x^{(0)}, x^{(1)}, \ldots, x^{(k)}$ in ascending order and denote them by $x_0, x_1, \ldots x_k$. That is, $a = x_0 \leq x_1 \leq \ldots \leq x_k = b$. Let

$$z_i = f(x_i) \text{ for } i = 0, 1, \ldots, k.$$

**1) Interval selection**

Let $R(i) = R(x_{i-1}, x_i, z_{i-1}, z_i)$ for $i = 1, 2, \ldots, k$.

Let $R(t) = \max_{1 \leq i \leq k} R(i)$. That is, select the interval $(x_{t-1}, x_t)$ from all intervals $(x_{i-1}, x_i)$ for $i = 1, 2, \ldots, k$ at which the interval-characteristic function attains its maximum.

**2) Point selection**

Let $S(t) = S(x_{t-1}, x_t, z_{t-1}, z_t) \in [a, b]$, the point selection function value at $(x_{t-1}, x_t, z_{t-1}, z_t)$. Then the $(k+1)$st evaluation point $x^{(k+1)}$ is determined by:

$$x^{(k+1)} = S(t).$$

For the above example of $R$, the interval-characteristic function value $R(i)$ is the negative minimum of the lower bounding function $F_k(x)$ on $[x_{i-1}, x_i]$, of the Piyavskii-Shubert algorithm:

$$R(i) = -\tfrac{1}{2}[M(x_{i-1} - x_i) + z_{i-1} + z_i].$$

If $t$ is the index at which $R$ attains its maximum, it is a point where the lower bounding function $F_k$ attains its minimum. The point selection function value, for the above example of $S$, will be

$$S(t) = \frac{1}{2M}[M(x_{t-1} + x_t) + z_{t-1} - z_t] = x^{(k+1)}$$

the global minimiser of the lower bounding function $F_k(x)$, of the Piyavskii-Shubert algorithm.

**Stopping Criterion:**

If a stopping criterion is met, then stop, otherwise increment $k$ and go to the next iteration. For example, the criterion might be that $x_{t+1} - x_t < \epsilon$, for $\epsilon > 0$, a preset tolerance.

**Axiomatic assumptions on the interval-characteristic function $R$ and the point selection function $S$:**

**A1.** $R(x_{i-1}, x_i, z_{i-1}, z_i)$ is a continuous function and $\lim_{k\to\infty} R(x_{i-1}, x_i, z_{i-1}, z_i)$ exists and is continuous.

**A2.** $R(x_{i-1}, x_i, z_{i-1}, z_i) = R(x_{i-1} + c, x_i + c, z_{i-1}, z_i)$ for any real number $c$ such that $x_{i-1} + c, x_i + c \in [a, b]$. In other words, $R$ is translation-invariant with respect to the interval $[x_{i-1}, x_i]$.

**A3.** $R(x_{i-1}, x_i, z_{i-1} - c_{i-1}, z_i - c_i) > R(x_{i-1}, x_i, z_{i-1}, z_i)$ for $c_{i-1} \geq 0, c_i \geq 0$; $c_{i-1}^2 + c_i^2 > 0$. In other words, $R$ is strictly decreasing with respect to the objective function values $z_{i-1}$ and $z_i$.

**A4.** $R(x_{i-1}, x_i, z_{i-1}, z_i) > R(x, x, f(x), f(x))$ for any $x \in [x_{i-1}, x_i]$. That is, the value of $R$ at a point with the first two variables distinct is strictly greater than the value of $R$ at any point with the first two variables the same and located between $x_{i-1}$ and $x_i$.

**A5.** $\max\{x^{(k+1)} - x_{t-1}, x_t - x^{(k+1)}\} \leq d(x_t - x_{t-1})$, where $1/2 \leq d < 1$. That is, the next evaluation point is not one of the end points $x_{t-1}$ or $x_t$ and is not too far away from the centre of the interval $[x_{t-1}, x_t]$.

**Convergence Results:**

If the interval selection function $R$ and the point selection function $S$ satisfy the above assumptions $A1$ to $A5$, then $A = E$, where $A$ is the set of accumulation

points of $\{x_0, x_1, \ldots, x_k\}$ and E is the set of global minimisers for problems $P$ and $Q$. This is proved in [52, pp.7-12] .

Pintér's method is a general axiomatic framework; many algorithms can be fitted into this framework. The Strongin algorithm [75] has been proved to be special cases of this axiom-based algorithm. The Piyavskii-Shubert algorithm is also proved to lie within this framework, under a mild condition: the used Lipschitz constant $M$ must be greater than the least Lipschitz constant. There is an example showing that this condition cannot be removed. That is, we can construct a function and with the use of the least Lipschitz constant, the Piyavskii-Shubert algorithm will result in $A \subseteq E$ and $A \neq E$. This will be discussed in detail in Chapter 4. The Žilinskas approach [85] does not fit completely into this framework, since it does not satisfy assumption A3 in limit. It generates, theoretically, an everywhere dense set of points on $[a, b]$. Pintér in [60] provides a more general convergence analysis and qualification of partition algorithms: that work fully subsumes, among others, the method of Žilinskas.

Pintér's method provides guidelines for researchers to construct their own algorithm. This can be done by defining the interval-characteristic function $R$ and point selection function $S$ according to the strategies of other algorithms and checking that the interval-characteristic function and point selection function satisfy assumptions $A1$-$A5$.

The one-dimensional version of Pintér's approach was generalized to the higher dimensional case by the same author in [55, 56], by changing the interval end points $x_i$ into the vertices of a high dimension interval $a_i \leq x_i \leq b_i$ for $i = 1, \ldots, d$. Lexicographical order is used to describe the vertices of such a higher dimensional interval. A diagonal interval selection function was discussed to reduce the computational complexity. It can be viewed as a direct extension of the one dimensional

case. Further significant generalizations are discussed by Pintér in [54] for convex optimisation, [58] for general Lipschitz programming and [60] necessary and sufficient convergence conditions for general convergence. It is remarkable that Pintér's Lipschitzian global optimisation approaches have been applied for solving a broad range of real-world problems. For instance, the methods have been applied to product design by Hendrix and Pintér, [34], to the calibration of nonlinear descriptive models by Pintér, Szabó and Somlyódy [57], Pintér [59] and to the black box design of engineering system by Boon, Pintér and Somlyódy [10].

We now review two multidimensional algorithms: the Mladineo algorithm [43] and the Wood algorithm [78]. These algorithms are multidimensional extensions of the Piyavskii-Shubert algorithm and are closely related to the work of this thesis.

### Mladineo algorithm

Mladineo proposed in [43] an algorithm for solving problem $P'$. It is a $d$-dimensional generalisation of the Piyavskii-Shubert algorithm. The lower bounding function in the higher dimensional case is made up from piecewise quadratic hypercones instead of the piecewise linear segments of the one dimensional case. This is a geometric consequence of the Lipschitz condition. Mladineo's main contribution is the provision of a mechanism for finding the actual minimum of the lower bounding function. This forms the location of the next evaluation point.

Let $D = I^d = \{x : 0 \leq x_i \leq 1, \ \ i = 1, \ldots, d\} \subset \mathbf{R}^d$, the unit hypercube, and take $f \in L(M)$ on $D$.

**Initialisation:**

The initial evaluation point $x_0$ is chosen arbitrarily in $D$. For instance $x_0$ can be chosen as the vertex $(0, 0, \ldots, 0)$. Evaluate $f$ at $x_0$ and let $f_*^0 = f(x_0)$.

Let the initial lower bounding function $F_0$ be determined by:

$$F_0(x) = f(x_0) - M||x - x_0||.$$

Set $k = 1$.

**Iterations:**

At the beginning of step $k$, we have a set of evaluation points $x_0, x_1, \ldots, x_{k-1}$, corresponding function values $f(x_0), f(x_1), \ldots, f(x_{k-1})$ and a lower bounding function $F_{k-1}(x)$. Let the $k$th evaluation point be:

$$x_k = \arg\min_{x \in D} F_{k-1}(x).$$

If the location $x_k$ is not unique, choose one arbitrarily from them. Evaluate $f$ at $x_k$. Then we have the least function value to date

$$f_*^k = \min_{0 \leq i \leq k} \{f(x_i)\}$$

and its location

$$x_*^k = \arg\min_{0 \leq i \leq k} \{f(x_i)\}.$$

If $x_*^k$ is not unique, choose one arbitrarily.

The $k$th lower bounding function $F_k$ is defined by

$$F_k(x) = \max_{0 \leq i \leq k} \{f(x_i) - M||x - x_i||\}.$$

**Stopping Criterion:**

Let $F_*^k = \min_{x \in D} F_k(x)$. If $f_*^k - F_*^k \leq \epsilon$, then stop. Otherwise increment $k$ and go to the next iteration.

**Convergence Results:**

$$1) \qquad \lim_{k \to \infty} f(x_k) = f_*,$$

$$2) \qquad \lim_{k \to \infty} \inf_{x \in E} ||x - x_k||) = 0.$$

The convergence result is proved in [43] by Mladineo. It is a *one-step optimal* algorithm. This means that the algorithm procedure minimises, over all sequential

algorithms, the maximum, over all functions which belong to $L(M)$ and pass through all evaluation points to date, the difference $f_*^k - F_*^k$. This is also proved in [43].

Mladineo recommended the use of the upper bound of the norm of the function's gradient to estimate the Lipschitz constant. A result of the convergence rates of her algorithm for a class of functions is presented in [44]. A stochastic variation of her algorithm is proposed in her later paper [45].

The Mladineo algorithm is a natural generalisation of the Piyavskii-Shubert algorithm to higher dimensions. The selection of a new evaluation point, however, involves solving all possible systems of $d$ linear equations and a quadratic equation. The number of such systems increases rapidly with the number of iterations. This affects the efficiency of the convergence, especially as dimension increases. More efficient ways for calculating the new evaluation point have been proposed by Strigul [73] and recently by Jaumard, Herremann and Ribault [37]. Strigul provides algebraic methods to avoid computing solutions of some systems which do not correspond to the global minimum of $F_k$ while Jaumard, Herremann and Ribault use a geometric argument to reduce the number of systems which need to be considered for obtaining the global minimum of $F_k$.

### Wood algorithm

Wood in [79] proposed an algorithm, called multidimensional bisection, for solving problems $P$ and $P'$. It can be viewed as a generalisation of the well-known bisection method in one dimension. It has been whimsically called the bee-section algorithm because of the shape the natural domain takes.

In terms of the the terminology of this section, the lower bounding function of the Wood algorithm can be expressed as a piecewise linear function. This makes the location of the deepest point easy to find. In fact, Wood constructed explicit

expressions for the coordinates of base points of simplexes in the system; these represent the local and global minima of the lower bounding function. The main virtue of this algorithm is the simple structure of the lower bounding function. It can be coded using a simple data structure. In [78], Wood proved the deepest point strategy of multidimensional bisection is a one-step optimal procedure.

In [86], acceleration methods and the performance of multidimensional bisection are investigated. In [5], Baritompa discusses multidimensional bisection using a geometric "dual" viewpoint and this idea is used to extend the Mladineo and Wood algorithms to a customising method for global optimisation in [6].

A detailed description of this algorithm, together with acceleration methods and performance details, will be presented in Chapter 4.

Many deterministic approaches to solving the global optimisation problem can be cast as branch and bound methods. We review Horst and Tuy's branch and bound framework in the next section. This framework is extended to the language of covers in Chapter 4. We then show that the Wood algorithm fits into this extended framework.

## 2.3 Horst and Tuy's branch and bound framework

A widely used framework for setting certain global optimisation methods is the branch and bound framework. Horst and Tuy in [35] give a formal description of a general branch and bound procedure. They also provide convergence conditions for this general framework. We concentrate on problems $P$, $P'$, $Q$ and $Q'$ in which the objective function satisfies a Lipschitzian assumption on its domain $D$. The procedure and the convergence conditions are stated now. This material is drawn from [35], and sets the scene for Section 4.4 in this thesis.

**Definition 2.3.1** *Let $D$ be a subset of $\boldsymbol{R}^d$ and $I$ be a finite index set. A set $\{B_i : i \in I\}$ of subsets of $D$ is said to be a partition of $D$ if $D = \cup_{i \in I} B_i$, and $B_i \cap B_j = \partial B_i \cap \partial B_j$ for all $i, j \in I$, $i \neq j$, where $\partial B_i$ denotes the boundary of $B_i$.*

It is natural to use polytopes or convex polyhedral sets as the partition sets. Figure 2.3 shows a partition of $D$ using simplexes for the case $d = 2$.

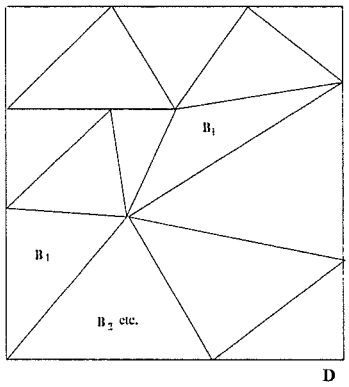



**Figure 2.3** A simplex partition of $D$ for the case $d = 2$.

If $B$ denotes an element of the current partition of $D$ used in a branch and bound procedure, it is important to be able to verify whether $B \cap D \neq \emptyset$. This allows us to determine whether to delete $B$ from the partition.

**Definition 2.3.2** *A partition set $B$ is called feasible if $B \cap D \neq \emptyset$, infeasible if $B \cap D = \emptyset$ and otherwise called uncertain (if we do not know whether it is feasible or infeasible).*

We now describe the branch and bound procedure for solving problem $P$ for an objective function $f$ on $D$:

**Step 0.** Choose a relaxed $B_0 \supseteq D$ and a feasible set $S_{B_0} \subset D$. Choose a real number $\beta_0$, an initial lower bound of $f$ on $S_{B_0}$, such that $-\infty < \beta_0 \leq \min f(D)$. Set the initial partition $\mu_0 = \{B_0\}$, an upper bound $\alpha_0 = \min f(S_{B_0})$ and a lower bound over $B_0$, $\beta(B_0)$ satisfying $\beta_0 = \beta(B_0) \leq \min f(D)$.

Let $x^0 = \text{argmin}\, f(S_{B_0})$ provided $\alpha_0 < \infty$. If $\alpha_0 - \beta_0 = 0$ then stop. The minimum of $f$ is $\min f(D) = \alpha_0 = \beta_0$, with the minimiser $x_* = x^0$.

Otherwise go to the next step.

**Step $k$.** At the beginning of step $k$, we have:

a) A partition $\mu_{k-1}$ of a subset of $B_0$ still of interest.

b) For all $B \in \mu_{k-1}$, there is an $S_B \subseteq B \cap D$, and bounds $\beta(B), \alpha(B)$ satisfying $\beta(B) \leq \inf f(B \cap D) \leq \alpha(B)$, if $B$ is feasible, and $\beta(B) \leq \inf f(B)$ if $B$ is uncertain. There are overall lower and upper bounds $\beta_{k-1}$ and $\alpha_{k-1}$ such that

$$\beta_{k-1} \leq \inf f(D) \leq \alpha_{k-1} \quad \text{and} \quad x^{k-1} \in D,$$

satisfying $f(x^{k-1}) = \alpha_{k-1}$.

**Sub-step $k1$.** Delete all $B \in \mu_{k-1}$ for which $\beta(B) \geq \alpha_{k-1}$. Denote by $R_k$ the remaining sets of $\mu_{k-1}$.

**Sub-step $k2$.** Select a non-empty collection of sets $P_k \subset R_k$, construct a partition of every set of $P_k$ and denote by $P_k'$ the collection of new partition sets.

**Sub-step $k3$.** Delete $B \in P_k'$ if $B \cap D = \emptyset$ or it is known that $\min f(D)$ cannot occur in $B$. Let $\mu_k'$ be the collection of all remaining members of $P_k'$.

**Sub-step $k4$.** Assign to each $B \in \mu_k'$ a set $S_B$ and a $\beta(B)$ such that $S_B \subseteq B \cap D, \beta(B) \leq \inf f(B \cap D) \leq \alpha(B)$ if $B$ is feasible and $\beta(B) \leq \inf f(B)$ if $B$ is uncertain. Set $\alpha(B) = \min f(S_B)$.

**Sub-step $k5$.** Set $\mu_k = (R_k - P_k) \cup \mu_k'$. Compute $\alpha_k = \inf\{\alpha(B) : B \in \mu_k\}$, $\beta_k = \min\{\beta(B) : B \in \mu_k\}$ and let $x^k \in D$ be such that $f(x^k) = \alpha_k$.

**Sub-step $k6$.** If $\alpha_k - \beta_k = 0$, then stop. The minimum of $f$ is $\min f(D) = \alpha_k = \beta_k$, with $x_* = x^k$.

Otherwise, go to step $k+1$.

The branch and bound procedures, sub-steps $k1 - k3$, are the branch procedures while sub-steps $k4 - k5$ are the bound procedures. In [35], a partition element

$B \in \mu_k$ is called fathomed if $\beta(B) \geq \alpha_{k-1}$, otherwise it is called unfathomed. Therefore, sub-step $k1$ eliminates the fathomed sets which are subsets of $B_0$ from the sets selected for further partition, as the global minimum cannot occur in them. The procedure selects sets $P_k$ from the remaining elements $R_k$ of partition $\mu_{k-1}$ for further partition. Figure 2.4 illustrates this procedure.

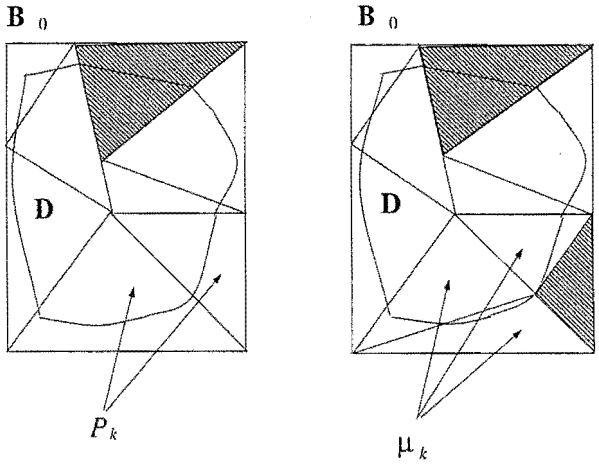



**Figure 2.4** A step of the branch and bound procedure where the shaded parts are fathomed partition elements.

To describe convergence conditions for branch and bound procedures, we need the following definitions.

**Definition 2.3.3** *A bounding operation is called consistent if at every step any unfathomed partition element can be further refined, and if any infinitely decreasing sequence* $\{B_{k_q}\}$ *of successively refined partition elements satisfies:*

$$\lim_{q \to \infty} (\alpha_{k_q} - \beta(B_{k_q})) = 0.$$

The consistent bounding operation insists that each uncertain subset which may contain the global minimum should be capable of refinement and that the difference between the lower bound of successively refined partition elements and the overall upper bound vanishes as the procedure goes on.

**Definition 2.3.4** *A selection operation is called complete if for every* $B \in \cup_{p=1}^{\infty} \cap_{k=p}^{\infty} R_k$ *we have*

$$\inf f(B \cap D) \geq \alpha = \lim_{k \to \infty} \alpha_k.$$

This definition means that any portion of the feasible set which is left unexplored forever must be not better than the fathomed portions.

**Definition 2.3.5** *A selection operation is called bound improving if at least each time after a finite number of steps,* $P_k$ *satisfies:*

$$P_k \cap arg\min\{\beta(B) : B \in R_k\} \neq \emptyset.$$

It means that at least one partition element where the actual lower bound is attained is selected for further partition in Step $k$ of the procedure.

**Definition 2.3.6** *A branch and bound procedure is called infinite if* $\alpha_k - \beta_k > 0$, *for every* $k$. *If there is a natural number* $k_0$ *such that* $\alpha_{k_0} - \beta_{k_0} = 0$, *the branch and bound procedure is called finite.*

Equipped with the above description of the general branch and bound procedure, convergence results can be proved. We state these results without proof. Proofs can be found in [35].

**Theorem 2.3.1** *In an infinite branch and bound procedure, suppose that the bounding operation is consistent and the selection operation is complete. Then*

$$\alpha = \lim_{k \to \infty} \alpha_k = \lim_{k \to \infty} f(x^k) = \min f(D).$$

**Theorem 2.3.2** *In a finite branch and bound procedure, suppose that the bounding operation is consistent and the selection operation is bound improving. Then the procedure is convergent:*

$$\alpha = \lim_{k \to \infty} \alpha_k = \lim_{k \to \infty} f(x^k) = \min f(D) = \lim_{k \to \infty} \beta_k = \beta.$$

The ideas behind branch and bound methods for solving global optimisation problems is straightforward. This procedure uses a partition of $D$ and bounds on $f$ to refine the search region and delete certain partition subsets of $D$. The framework presented is a rather general framework. In [36], many known Lipschitz based algorithms are proved to belong to this framework. Such are the Pintér method, the approach of Galperin and Zheng [24], the algorithms of Piyavskii-Shubert and that of Mladineo. We will show in Chapter 4 that in an enlarged framework Wood's algorithm also belongs to a branch and bound family.

All the deterministic algorithms based on the Lipschitz assumption for the objective function have a major drawback: the number of function evaluations required to convergence increases exponentially with the dimension $d$. In order to show this, it suffices to consider the operation of a Lipschitz based deterministic algorithm $A$ on the constant function, $f(x) = c$, over domain $D$, say the hypersphere with fixed radius $r$. Denote by $\mu(D)$ the Lebesgue measure of $D$, then

$$\mu(D) = \frac{\pi^{d/2} r^d}{\Gamma(1 + d/2)}$$

where $\Gamma$ denotes the gamma function.

Suppose that algorithm $A$ is applied to solve problem $P'$ over $D$ with radius $r = 1$ for this function and uses a Lipschitz constant $M > 0$. For an evaluation at a point $x \in D$, we know surely for any point $z$ in $\{z \in D : ||z - x|| \leq \epsilon/M\}$, a sphere around $x$, that $f(z)$ is within $\epsilon$ of $f(x)$. In order to know that the value $c$ is within $\epsilon$ of $y_*$, the unknown global minimum of $f$, we must cover $D$ with such spheres.

Suppose that $f$ has been evaluated by $A$ in $k$ points $x_1, x_2, \ldots, x_k$, such that the union of the spheres $C_i = \{z \in D : ||z - x_i|| \leq \epsilon/M\}$, $i = 1, \ldots, k$ covers $D$. Then we have

$$\mu(D) \leq \frac{k(\frac{\epsilon}{M})^d \pi^{d/2}}{\Gamma(1 + d/2)}.$$

Thus, for the $k$ hyperspheres to cover $D$ we require

$$k\left(\frac{\epsilon}{M}\right)^d \pi^{d/2} > \frac{\pi^{d/2}}{\Gamma(1+d/2)}$$

which leads to

$$k > (\frac{M}{\epsilon})^d.$$

Choose $\epsilon < M$. The computational effort required then increases exponentially with dimension $d$.

A natural question can be raised: is there any stochastic method which can eliminate this convergence inefficiency? We review two stochastic methods for solving problem $P'$ in the next section. The second method, pure adaptive search, analysed by Zabinsky and Smith, gives a desirable convergence property. Pure adaptive search is a theoretical search approach, for which it has been proved that the number of iterations to convergence increases linearly with dimension. At first glance this appears to contradict the earlier statement about exponential increase of function evaluations with dimension. Note, however, that iterations in pure adaptive search are not comparable to function evaluations in a deterministic Lipschitz algorithm. Also, the pure adaptive search result is for the expected number of iterations, not the worst case number of function evaluations of the deterministic Lipschitz case.

## 2.4 Stochastic methods

Stochastic methods for solving global optimisation problems involve the evaluation of $f$ at randomly sampled points from $D$. Convergence with probability one to an $\epsilon$ solution has been proved, provided the sampling distribution and objective function $f$ satisfy certain conditions. For instance, see [4], [9], [14], [15], [40], [51], [53] [87] and [72]. Conditions on $f$ and $D$ ensuring $\epsilon$-convergence can be that the objective function $f$ be a measurable function on a measurable domain $D$, and the sampling

distribution must satisfy

$$\Pi_{k=0}^{\infty}[1 - \mu_k(A)] = 0,$$

where $A$ is a Borel subset of $D$ and $\mu_k(\cdot)$ are a probability measure on $\mathbf{R}^d$ corresponding to the distribution functions used at step $k$ of an algorithm.

In this section, we review two stochastic methods, pure random search and pure adaptive search, which have close connections with the pure localisation search introduced in Chapter 5.

### Pure random search

Pure random search (PRS) is a simple stochastic method for solving problem $P'$. The following describes pure random search with a uniform distribution.

Let $D \subset \mathbf{R}^d$ be the measurable domain of the objective function $f$.

**Step 0.** Generate $X_0$ uniformly on $D$ and evaluate $f(X_0) = Y_0$.

Set $k = 1$.

**Step 1.** Generate $X_k$ uniformly on $D$ and evaluate $f(X_k) = Y_k$.

Let $Y_*^k = \min\{Y_0, \ldots, Y_k\}$.

**Step 2.** If a stopping criterion is met, let $f_{opt} = Y_*^k$ and $x_{opt} = \text{argmin}\{Y_0, \ldots, Y_k\}$. Otherwise increment $k$ and return to Step 1.

Pure random search is usually stopped at a fixed iteration number.

Brooks [11] and Anderssen [1] discussed the properties of pure random search. Pure random search is a passive method, as it does not use the previous evaluation results to update the procedure. It has the virtue of simplicity if the domain $D$ is in a regular form, for example, a hyper-rectangle in $\mathbf{R}^d$. The main difficulty with PRS is the generation of a random point in a non-regular region $D$. Even for the uniform distribution, this can be difficult to implement efficiently. For example, see [71]. Also, PRS is slow to converge. The difficulty in implementing pure random

search is of a similar scale to that for pure adaptive search.

### Pure adaptive search

Zabinsky and Smith in [81] analysed a stochastic algorithm for solving problem $P'$, called pure adaptive search (PAS). This algorithm proceeds by generating a sequence of points uniformly distributed in nested regions of the feasible space. At any iteration, the next point in the sequence is uniformly distributed over the strictly improving set. This is the set on which the function value is strictly less than the current minimum.

Let the domain of the objective function $f$ be $D$ and $X_i$ be the $i$th evaluation point (considered as a random variable), $W_i = f(X_i)$ and $y^* = \max_{x \in D}\{f(x)\}$.

**Step 0.** Let $S_0 = D$ and $W_0 \geq y^*$. Set $k = 1$

**Step 1.** Generate $X_k$ uniformly in $S_k = \{x \in S_0 : \ f(x) < W_{k-1}\}$.

**Step 2.** Set $W_k = f(X_k)$. If a stopping criterion is met, stop, and let $f_{opt} = W_k$ and $x_{opt} = X_k$. Otherwise increment $k$ and return to Step 1.

The most remarkable result of the PAS algorithm for solving problem $P'$ is that the complexity, measured by the expected number of iterations to convergence, increases at most linearly with the dimension of the problem, provided that the objective function has a fixed Lipschitzian bound and the diameter of $D$ is fixed. That is,

$$E\left[N^*_{PAS}(y)\right] \ \leq \ 1 + d\delta \ln(Mr/(y - y_*),$$

where $N^*_{PAS}(y)$ is the expected number of iterations for PAS to achieve a value of $y$ or lower, $d$ is the dimension of the domain $D$ and $\delta$ is the fixed diameter of $D$. This result hints at the existence of good random search methods and suggests a combination of deterministic methods and stochastic methods could be useful for solving global optimisation problems.

As mentioned by Zabinsky and Smith, it is difficult to implement PAS efficiently. The main reason for this is that the geometric description of an arbitrary level set is unknown. Furthermore, even if we did know the level set, there is no known efficient procedure for generating a single point which is uniformly distributed in a general region. In [7], a general framework is proposed which generalises PAS to "somewhat adaptive search" and introduces a new algorithm which attempts to reach the practical ideal of the linearity in dimension result of PAS. We will discuss PAS and its generalisation, "somewhat adaptive search" and $\rho$-adaptive search, in Chapter 5. pure localisation search, a combination of the Piyavskii-Shubert algorithm and PAS ideas is also investigated in Chapter 5.

CHAPTER 3

# Convergence properties of Lipschitz based algorithms

## Introduction

In this chapter, we discuss some general convergence properties of Lipschitz based algorithms. Notice that the assumption of Lipschitz continuity of an objective function $f$ specifies that the absolute slope of the segment joining any two points on the graph of $f$ in the domain $D$ must be bounded above by a positive constant $M$. The effect an algorithm has, therefore, is to reduce the size of the uncertain region $U_k$, where $U_k \subset \mathbf{R}^{d+1}$ is the bracket after $k$ evaluations of $f$ as described in Section 2.1. Thus $U_k$ gives us the best current bounds for both the global minimum of $f$ and the localisation. It is evident that the availability of the Lipschitz constant is critical in any Lipschitz based algorithm. Lipschitzian global optimisation has inherent difficulty finding an exact minimum of a function $f$. Thus a Lipschitz based algorithm rarely solves problems $P$ and $Q$ in a finite number of function evaluations.

In Section 3.1 we confirm that $F_k$, the lower bounding function of $f$ introduced in Section 2.1, is a best Lipschitzian lower bounding function after a set of evaluations for a Lipschitz based algorithm. In Section 3.2 we present a necessary condition on the objective function for finite convergence of a Lipschitz based algo-

rithm. These are extensions of the one dimensional results of Hansen, Jaumard and Lu discussed in [29], to the higher dimensional case. Note that the finite convergence problem and related results are largely of theoretical interest.

## 3.1 The tightest bounding function

Any Lipschitz based algorithm for solving global optimisation problems can use this property of the objective function to reduce the size of bracket $U_k \subset \mathbf{R}^{d+1}$ (Section 2.1). The lower envelope of a bracket can form a function which bounds the function $f$ from below. We give the following definition.

**Definition 3.1.1** *For functions $F$ and $f$ defined on $D$, if $F(x) \leq f(x)$ for all $x \in D$, we say that $F$ is a* lower bounding function *of $f$ over $D$.*

For an objective function $f \in L(M)$ over $D$, after evaluating $f$ at $k$ points $x_1, x_2, \ldots, x_k$ on $D$, a natural Lipschitzian lower bounding function which belongs to $L(M)$ and has the same function values at $x_i$, for $1 \leq i \leq k$, is the function $F_k(x) = \max_{1 \leq i \leq k}\{f(x_i) - M||x - x_i||\}$ over $D$. This is a "best" Lipschitzian lower bounding function of $f$ in the following sense.

**Definition 3.1.2** *For given $f \in L(M)$ and points $x_1, x_2, \ldots, x_k$ on $D$, a function $F$ on $D$ is called the* tightest Lipschitzian lower bounding function *of $f$ if*

*1) $F \in L(M)$ and $F(x_i) = f(x_i)$, for $i = 1, 2, \ldots, k$;*

*2) For any $g \in L(M)$ such that $g(x_i) = f(x_i)$ for $i = 1, 2, \ldots, k$ $F \leq g$ on $D$.*

It is easy to see that $F_k$ defined above is the tightest Lipschitzian lower bounding function in the light of the fact that only the function values at $x_i$, for $i = 1, 2, \ldots, k$, are known. It is called a "saw-tooth" cover in [29] for the case of $d = 1$, because of its shape.

**Proposition 3.1.1** *For $f \in L(M)$ over $D \subset \boldsymbol{R}^d$, after evaluating the function $f$ at $k$ points $x_1, x_2, \ldots, x_k$ on $D$, the tightest Lipschitzian lower bounding function $F_k$ for $f$ on $D$ is:*

$$F_k(x) = \max_{1 \le i \le k} \{ f(x_i) - M||x - x_i|| \}.$$

Proof: Firstly, it is readily shown that $F_k \in L(M)$ and $F(x_i) = f(x_i)$, for $i = 1, 2, \ldots, k$.

Secondly, we show that for any lower bounding function $g$ of $f$ such that $g \in L(M)$ and $g(x_i) = f(x_i)$ for $i = 1, 2, \ldots, k$, we have $F_k \le g$. If this were not the case, there would exist $z \in D$, such that $g(z) < F_k(z)$. From the above definition of $F_k$, we have that $z \neq x_i$, for all $i = 1, 2, \ldots, k$. Choose an evaluation point $x_t$ from $x_1, x_2, \ldots, x_k$ such that $F_k(z) = g(x_t) - M||z - x_t||$. Then $g(z) < F_k(z) = g(x_t) - M||z - x_t||$, and therefore $g(x_t) - g(z) > M||z - x_t||$, a contradiction. □

The Lipschitzian lower bounding function $F_k$ is a natural generalisation of the Piyavskii-Shubert lower envelope to higher dimensions. It was first used as a Lipschitzian lower bounding function by Mladineo in her algorithm [43]. As this generalisation gives a nonlinear bound instead of the piecewise linear bound of dimension one, it is harder to determine the minimum of $F_k$, used as the next evaluation point for her algorithm.

## 3.2 Necessary conditions for finite convergence

Can an algorithm solve problem $P$ or $Q$ in a finite number of steps? That is, can we find an exact global minimum or the exact global minimiser(s) of a Lipschitz function $f$ over the domain $D$ with just a finite number of function evaluations? We investigate this now. Generally speaking, finite convergence for problem $P$ or $Q$ imposes very strict conditions on both the algorithm and the objective function. In fact, as Hansen Jaumard and Lu proved, for the case of one dimension, unless

the objective function has a "$V$" shape at the minimiser with the slope exactly the least Lipschitz constant, no algorithm can solve $P$ or $Q$ in finitely many steps. Here we prove similar results (Theorem 3.2.1 and 3.2.2) for the higher dimensional case. We need the following lemma.

**Lemma 3.2.1** *Let $f \in L(M)$ have compact convex domain $D \subset \boldsymbol{R}^d$. If there exist $x_1, x_2 \in D$, such that $f(x_2) - f(x_1) = M||x_2 - x_1||$, then $f(x) - f(x_1) = M||x - x_1||$ for all $x = \lambda x_2 + (1-\lambda)x_1$, where $\lambda \in [0,1]$.*

Proof: Since $f(x_2) = f(x_1) + M||x_2 - x_1||$, we have $f(x_2) \geq f(x_1)$. If there exists an $x_0 = \lambda_0 x_2 + (1-\lambda_0)x_1$, for some $\lambda_0 \in [0,1]$, such that $f(x_0) \neq f(x_1) + M||x_0 - x_1||$, (note that this assumption implies $\lambda_0 \neq 0, 1$ in accordance with the conditions of the Lemma), then we would have either

$$f(x_0) > f(x_1) + M||x_0 - x_1|| \quad \text{or} \quad f(x_0) < f(x_1) + M||x_0 - x_1||.$$

In the first case we would have $f(x_0) > f(x_1)$ and hence

$$|f(x_0) - f(x_1)| = f(x_0) - f(x_1) > M||x_0 - x_1||.$$

In the second case, we would have

$$\begin{aligned} & M||x_2 - x_0|| \\ = \; & M||x_2 - x_1|| - M||x_1 - x_0|| \\ = \; & f(x_2) - f(x_1) - M||x_1 - x_0|| \\ < \; & (f(x_2) - f(x_1)) + (f(x_1) - f(x_0)) \\ = \; & f(x_2) - f(x_0). \end{aligned}$$

Both contradict the assumption that $f \in L(M)$.

□

We need the following notation to describe our theorems. For a given function $f \in L(M)$, after evaluations of $f$ at $x_1, x_2, \ldots, x_k$ we have a Lipschitzian lower bounding function

$$F_k(x) = \max_{1 \le i \le k} \{f(x_i) - M||x - x_i||\}.$$

Let $f_* = \min_{x \in D} f(x)$ and $x_* = \text{argmin}_{x \in D} f(x)$. Let $f_*^k = \min_{1 \le i \le k}\{f(x_i)\}$ be the current least function value and $x_*^k = \arg\min_{1 \le i \le k}\{f(x_i)\}$, be the location of the current minimum. Let $\hat{E} = \{\hat{x} : \hat{x} = \text{argmin}_x F_k(x)\}$, the collection of all global minimisers of $F_k$ over $D$. Let $\hat{x} \in \hat{E}$ be an arbitrary global minimiser of $F_k$. We state the following facts without proof.

**Fact 1** If $F_k(x) < f_*^k$ then $x \notin \{x_1, \ldots, x_k\}$. Thus, the lowest point of $F_k$ does not occur at a peak point of $F_k$.

**Fact 2** If $\hat{x}$, an arbitrary global minimiser of $F_k$, lies in $\text{int}D$, the interior of $D$, then there exist $d+1$ hypercones with apexes at $y_1, y_2, \ldots, y_{d+1} \in \{x_1, \ldots, x_k\}$ such that $y_1, y_2, \ldots, y_{d+1}$ are affinely independent and $F_k(\hat{x}) = \min_{x \in D} \max_{1 \le i \le d+1}\{f(y_i) - M||x - x_i||\}$, the lowest point of $F_k$ over $D$. See Figure 3.1 .

**Fact 3** If $\hat{x} \in \partial D$, the boundary of $D$, then there exist $m$ hypercones ($m < d+1$) with apexes at $y_1, y_2, \ldots, y_m \in \{x_1, \ldots, x_k\}$ such that $y_1, y_2, \ldots, y_m$ are linearly independent and $F_k(\hat{x}) = \min_{x \in D} \max_{1 \le i \le m}\{f(y_i) - M||x - x_i||\}$, the lowest point of $F_k$ over $D$.

**Fact 4** The set $\hat{E}$ has a finite number of elements and $F_k(x) > F_k(\hat{x})$ for any point $x \in D - \hat{E}$.

From the above observations, an algorithm will terminate with convergence to an exact global minimum of $f$ in a finite number of steps only if the global minimiser $x_*$ of $f$ happens to be $\hat{x}$ for some $k$. This suggests necessary conditions upon the objective function $f$ to ensure finite convergence. We describe these as follows:

**Condition** $C_1$ : There exist $y_1, y_2, \ldots, y_{d+1} \in D$, affinely independent, such that $f(x) = f(x_*) + M||x - y_i||$, for all $x = \lambda x_* + (1-\lambda)y_i$, where $\lambda \in [0,1]$ and $i = 1, 2 \ldots, d+1$.

**Condition** $C_2$ : There exist $y_1, y_2, \ldots, y_m \in D(m < d+1)$, linearly independent, such that $f(x) = f(x_*) + M||x - y_i||$, for all $x = \lambda x_* + (1-\lambda)y_i$, where $\lambda \in [0,1]$ and $i = 1, 2 \ldots, m$.

Let $\Phi = \{f \in L(M) : f$ satisfies $C_1$ if $x_* \in \text{int}D$ and $f$ satisfies $C_2$ if $x_* \in \partial D\}$.

A function $f \in \Phi$ if there are $d+1$ affinely independent directions ($m < d+1$, linearly independent, if $x_*$ is on the boundary) over which $f$ coincides with the hypercones $y = f(x_*) + M||x - y_i||$ in a neighbourhood of $x_*$. In other words, there exist $d+1$ ($m$ if $x_*$ is on the boundary) removal hypercones which intersect at $x_*$. Figure 3.1 illustrates the case for $d = 2$.

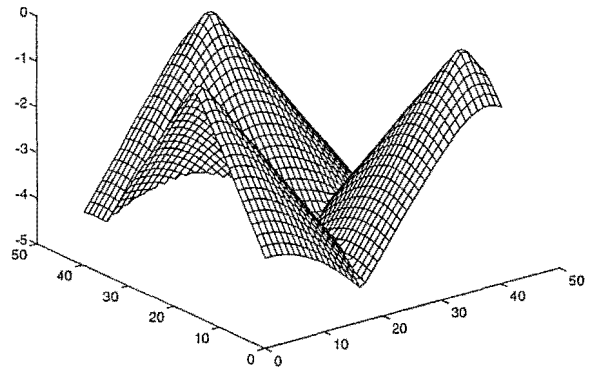



**Figure 3.1** Local behaviour of a function $f$ in $\Phi$ at a global minimiser $x_*$ of $f$.

With the above definitions, we are ready to give the following theorem.

**Theorem 3.2.1** *There is no sequential Lipschitz based algorithm that can solve problem $P$ using only a finite numbers of function evaluations, unless $f \in \Phi$.*

Proof: Suppose that $f \notin \Phi$ and there is a sequential Lipschitz based algorithm $A$ that solves problem $P$ in $k$ steps. We would have evaluation points $x_1, x_2, \ldots, x_k$,

and a $z$ such that

$$f(z) = f_* = \min_{x \in D} f(x).$$

We can therefore construct aLipschitzian lower bounding functionn

$$F_k(x) = \max_{1 \le i \le k} \{f(x_i) - M||x - x_i||\}.$$

We consider two cases:

*Case 1:* The point $z$ belongs to the interior of $D$. For this case we must have $k \geq d+1$. Recall that $\hat{E}$ is the set of minimisers of the Lipschitzian lower bounding function $F_k$. Again we consider two cases.

i) $z \notin \hat{E}$ : From the construction of $F_k(x)$ and Fact 4,

$$f(z) \geq F_k(z) > F_k(\hat{x}) \text{ for any } \hat{x} \in \hat{E}.$$

Let $g = F_k$ on $D$. Then $g \in L(M)$, $g(x_i) = f(x_i)$ for $i = 1, \ldots, k$ and

$$\min_{x \in D} g(x) = F_k(\hat{x}), \qquad \text{for some } \hat{x} \in \hat{E},$$
$$< f(z) = f_* = \min_{x \in D} f(x).$$

Figure 3.2 illustrates this case for $d = 1$.

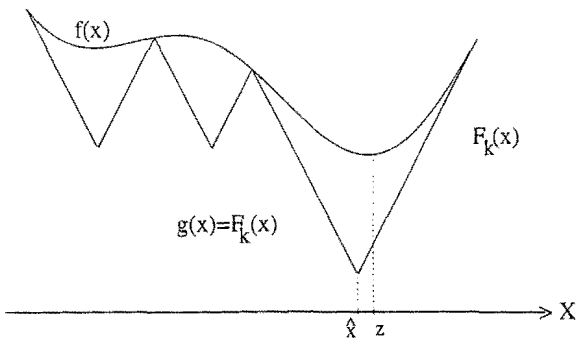



**Figure 3.2** The construction of $g$ for the case when $z$ is not a global minimiser of $F_k$.

ii) $z \in \hat{E}$ : Since $f \notin \Phi$, $f(x) > F_k(x)$ for $x \in \hat{E}$, otherwise there would be $d+1$ affinely independent evaluation points, for which $f$ satisfies $C1$, a contradiction.

Thus

$$f(z) > F_k(z).$$

Define $g$ over $D$ via

$$g(x) = \max_{x \in D}\{F_k(x), f(z) - M||x - z||\}.$$

That is, $g$ is a Lipschitzian lower bounding function passing through $(x_1, f(x_1)), \ldots, (x_k, f(x_k)$ and $(z, f(z))$. From Fact 4,

$$\min_{x \in D} g(x) < g(z) = f(z).$$

Figure 3.3 illustrates this case for $d = 1$.

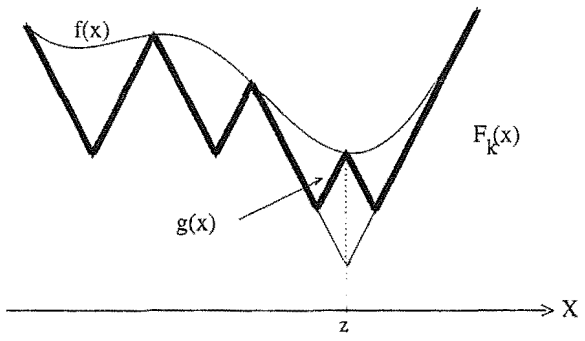



**Figure 3.3** The construction of $g$ for the case when $z$ is a global minimiser of $F_k$.

Now all information obtained on $f$ and $g$ by algorithm $A$ is the same after $k$ steps. That is, the evaluation points $x_1, \ldots, x_k$ and corresponding function values are the same, for $i = 1, 2, \ldots, k$. Because $A$ is a sequential algorithm, the decision on the next evaluation point and conclusion only depend on the above information and the Lipschitz constant $M$ used. So $A$ will claim that $g(z)$ is the global minimum of function $g$, together with minimiser $z$ as it does for function $f$, a contradiction.

*Case 2:* The claimed minimiser $z$ belongs to the boundary of $D$.

The proof for this case is similar to Case 1. If there is an algorithm which solves problem $P$ for $f$ in a finite number of steps, we can construct another function $g$ with the same information as that for $f$ after $k$ steps, but with a different minimiser and a smaller minimum. Algorithm $A$ will claim that $f(z)$ is the global minimum of function $g$, together with the same minimiser, which is a contradiction. □

From the above theorem, it is easy to show that:

**Corollary 3.2.1** *There is no sequential Lipschitz based algorithm that can solve problem $Q$ using only a finite numbers of function evaluations, unless $f \in \Phi$.*

Proof: If there is an algorithm $A$ that solves problem $Q$ for $f \notin \Phi$ using a finite numbers of function evaluations, then it would also solve problem $P$. This contradicts the conclusion of Theorem 3.2.1. □

The following proposition shows that the requirement that $f \in \Phi$ is not a sufficient condition for a sequential algorithm to converge in finitely many steps.

**Proposition 3.2.1** *There is no sequential Lipschitz based algorithm that can solve problem $P$ or $Q$ using only a finite numbers of function evaluations, unless it uses the exact value of the Lipschitz constant $M$ to bound the function $f$.*

The proof is similar to that of Theorem 3.2.1. If an algorithm $A$ used a larger Lipschitz constant $M_0$ than the exact one, $M$, and claimed a global minimiser of $f$ after a finitely number of evaluations, we can then construct another function $g \in L(M)$ such that $g$ has the same function values at the finitely many evaluation points to date for function $f$, but a different minimiser and smaller minimum. Algorithm $A$ then fails to find the right minimum of $g$. We omit the detailed proof of this proposition.

The following theorem shows the difficulties associated with locating the global optimum.

**Theorem 3.2.2** *No sequential Lipschitz based algorithm can localise a global minimiser in a compact subset $D^*$ of $D$, using only a finite number of function evaluations, unless $D^* \supseteq S$, where $S = \{x \in D : F_k(x) \leq f_{opt}\}$, the level set of the lower bounding function, where $F_k(x) = \max_{1 \leq i \leq k}\{f(x_i) - M||x - x_i||\}$, $f_{opt} = \min_{1 \leq i \leq k}\{f(x_i)\}$, and $x_i$ for $i = 1, 2, \ldots, k$ are the evaluation points of $f$.*

Proof: Suppose there is a sequential Lipschitz based algorithm $A$ which is able to localise a global minimiser $x_*$ in a compact subset $D^*$, such that $D^* \subseteq S$ and $D^* \neq S$, after evaluating function $f$ at the $k$ points $x_1, x_2, \ldots, x_k$. Because both $D^*$ and $S$ are compact, $S - D^*$ is non-empty. Therefore, there is a closed ball $B(x', \delta)$ with positive radius $\delta$ and centre at $x'$, such that $B(x', \delta) \subset S - D^*$. Figure 3.4 illustrates this for $d = 2$.

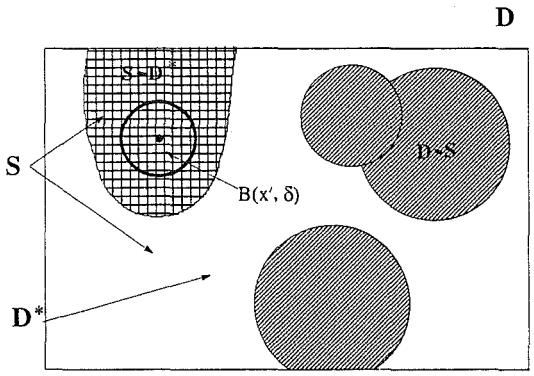



**Figure 3.4** Partition of domain $D$ of function $g$, where the blank and grid region $S$ is the level set of the Lipschitzian lower bounding function $F_k$ and the blank region $D^*$ is a compact subset of $D$.

Define a function $g$ over $D$ as follows:

$$g(x) = \begin{cases} F_k(x) & \text{if} \quad x \in D - S \\ f_{opt} & \text{if} \quad x \in S - \overline{B(x', \delta)} \\ c(x) & \text{if} \quad x \in \overline{B(x', \delta)} \end{cases}$$

where $c(x)$ is a hypercone over the region $B(x', \delta)$ which has apex at $(x', f_{opt} - M\delta)$ and has intersection with hyperplane $y = f_{opt}$ on the boundary of $B(x', \delta)$.

It is easy to check that $g \in L(M)$ and $g(x_i) = f(x_i)$ for $i = 1, 2, \ldots, k$. Therefore, applying the sequential algorithm $A$ to both functions $f$ and $g$ yields exactly the same sequence of evaluation points $x_1, x_2, \ldots, x_k$. So algorithm $A$ also localises the global minimiser $x_*$ for $g$ within $D^*$. However, $\min_{x \in D} g(x) = \min_{x \in (D - D^*)} g(x) = f_{opt} - M\delta < f_{opt}$, which means that $D^*$ does not include the minimiser of $g$, a contradiction. □

Theorem 3.2.2 implies that problem $Q'$ may require an arbitrarily large number of function evaluations.

In practice, we will consider problems $P'$ and $Q'$ only and will not seek the exact solution as in $P$ and $Q$. Problems $P$ and $Q$ are investigated mainly for theoretical interest.

CHAPTER 4

# A deterministic method: multidimensional bisection

The Wood algorithm outlined in Chapter 2 is a generalisation of the familiar "interval-halving" bisection method to higher dimensions. The algorithm can be viewed as a mechanism for finding all the global minima of a real-valued Lipschitz continuous function over a natural compact domain in $\mathbf{R}^d$.

In this chapter, a formal description of multidimensional bisection and related convergence results are presented in Section 4.1. In Section 4.2 we discuss the acceleration of multidimensional bisection. Two acceleration methods are considered. Together they bring the algorithm closer to that of Mladineo, while retaining the simplicity of the simplex-based multidimensional bisection algorithm. We then explore two aspects of these multidimensional bisection algorithms: their numerical performance, and their relationship to branch and bound algorithms. In Section 4.3 we address the specific question "How do the algorithms perform?" Numerical results are presented for five test functions, three of them are test functions in the literature while the other two are drawn from a non-differentiable family of test functions. A comparison of the number of function evaluations of raw multidimensional bisection and its accelerations for six Hansen-Jaumard test functions adapted from [28] is also presented. In Section 4.4 a modification of the branch and bound

framework of Horst and Tuy, reviewed in Chapter 2, is given. We show that multidimensional bisection algorithms are included in a modification of the branch and bound framework in Section 4.5. In Section 4.6 the approach of Basso [8] is discussed for solving problem $Q$. A result which simplifies and extends a result of Basso, a strategy for choosing evaluation points, is presented which ensures the convergence of localisations. Many of the results in this chapter have appeared in [86].

## 4.1 Review of multidimensional bisection

We proceed to give a detailed description of the multidimensional bisection algorithms in this section. A set of figures is presented to illustrate the situation for the two dimensional case.

Let $\{u_1, \ldots, u_{d+1}\}$ comprise the unit vectors from the origin to the vertices of a regular simplex, with centroid the origin, in $\mathbf{R}^d$. Thus $u_1 + \ldots + u_{d+1} = 0$ and $u_k \cdot u_l = -1/d$ for all distinct pairs $k$ and $l$. Let $\bigtriangledown$ be the cone in $\mathbf{R}^{d+1}$ with apex the origin and cross-section $\mathrm{co}\{u_1, \ldots, u_{d+1}\}$ at height $M$ along the $(d+1)$st axis, where "co" denotes the convex hull. Formally,

$$\bigtriangledown \quad = \quad \mathrm{pos}\{(u_k, M) \ : \ k = 1, \ldots, d+1\},$$

where "pos" denotes all positive linear combinations. See Figure 4.1.

The following concepts are used to describe the algorithm.

**Definition 4.1.1**

1. *A* standard simplex *in* $\boldsymbol{R}^{d+1}$ *is a translate of a cap of the cone* $\bigtriangledown$*, so has the form*

$$T(x,y,h) \quad = \quad \mathrm{co}\left\{(x,y), (x + \frac{h}{M}u_k, y + h) : k = 1, \ldots, d+1\right\},$$

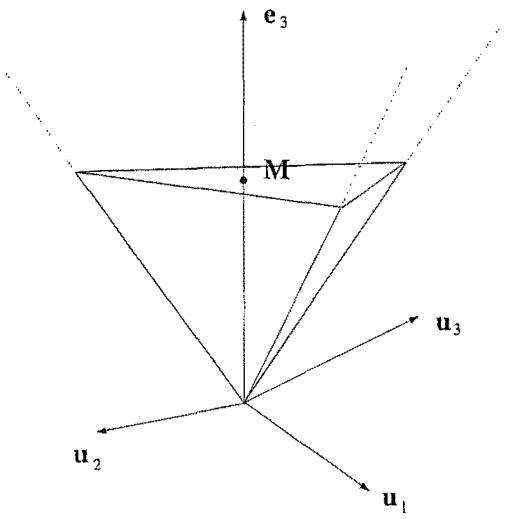



**Figure 4.1** The simplex ordering cone $\bigtriangledown$ with $M$, the slope along the edges.

*where* $(x, y) \in \boldsymbol{R}^{d+1}$ *is the apex, and* $h \in \boldsymbol{R}$ *the height. By the* top *of* $T(x, y, h)$ *is meant the facet of* $T(x, y, h)$ *opposite the apex.* See Figure 4.2.

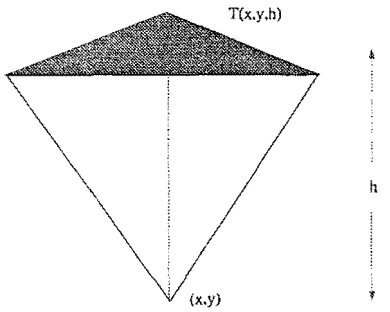



**Figure 4.2** A standard simplex $T(x, y, h)$ with top shaded.

2. *A* system of simplexes (*or* system) $\mathcal{S}$ *in* $\boldsymbol{R}^{d+1}$ *is*

   a) *a finite set* $\mathcal{T}$ *of standard simplexes, and*

   b) *a point* $a$ *in* $\boldsymbol{R}^{d+1}$, *lying in a lowest top of the system.*

3. *A* uniform system *is a system* $\mathcal{S}$ *in which all tops lie in the same hyperplane of* $\boldsymbol{R}^{d+1}$. *Or,* $y_j + h_j = y_k + h_k$ *for all* $T_j$, $T_k$ *in* $\mathcal{T}$. See Figure 4.3.

*4. The* variation *of a system $\mathcal{S}$, $V(\mathcal{S})$, is the difference between the highest and lowest points in the system. That is,*

$$V(\mathcal{S}) = \max_j\{y_j + h_j\} - \min_j\{y_j\}.$$

This idea is illustrated in Figure 4.3.

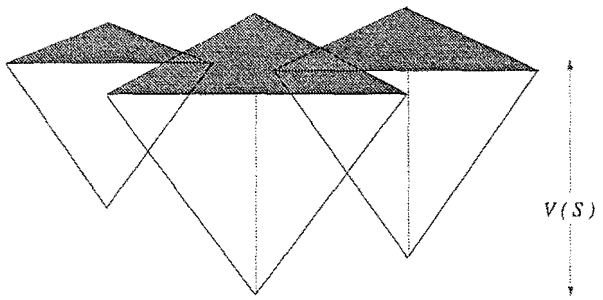



**Figure 4.3** A uniform system: overlapping standard simplexes forming a bracket for the global minima, the situation at the end of each iteration.

*5. A* standard domain *in $\boldsymbol{R}^d$ has the form $c + rU$, where $c \in \boldsymbol{R}^d$, $r \geq 0$ and*

$$U = \{x \in \boldsymbol{R}^d : x = \sum_{k=1}^{d+1} \lambda_k u_k, \quad 0 \leq \lambda_k \leq 1 \quad \textit{for} \quad k = 1, 2, ..., d+1\}$$

*A standard domain is a regular convex set which is centred at c and consists of all convex combinations of the unit vectors $u_1, \ldots, u_{d+1}$.*

A standard domain is a line segment when $d = 1$, a hexagon when $d = 2$ and a rhombic dodecahedron, the honeycomb cell, when $d = 3$. Figure 4.4 shows the case for $d = 2$. The standard domain is required for the setting up of an initial system. An initial system, described next, will then bracket all global minima of the objective function over a standard domain.

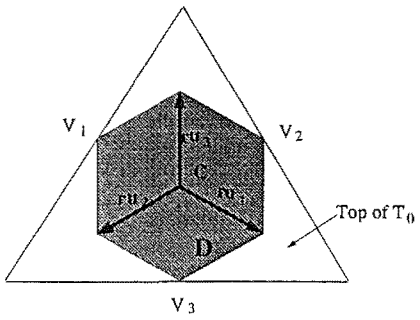



**Figure 4.4** Standard domain $D$ and the top of initial simplex $T_0$ for the multidimensional bisection algorithm for the case $d = 2$.

### The initial system

Given a standard domain, $D = c + rU$, an initial standard simplex, $T_0$, which brackets all global minima over $D$, can be set up as follows:

i) evaluate $f$ at the $d+1$ "dual" vertices of $D$, $\{v_k = c - ru_k \ : \ k = 1, \ldots, d+1\}$,

ii) remove the cones $(v_k, f(v_k)) - \nabla$ from $\mathbf{R}^{d+1}$, for each $k$, and

iii) truncate $\mathbf{R}^{d+1}$ at the level of the lowest evaluation.

This process yields an initial simplex in a natural way. This initial system can be written in terms of a simplex:

**Definition 4.1.2** *For $D = c + rU$, a standard domain in $\boldsymbol{R}^d$, and function $f$ in $L(M)$, the initial system $\mathcal{S}_0 = (\mathcal{T}_0, a_0)$ with $\mathcal{T}_0 = \{T_0 = T(x_0, y_0, h_0)\}$ is given by*

$$\begin{aligned} x_0 &= c + \frac{1}{M(d+1)} \sum_{k=1}^{d+1} (f(v_k) - m) u_k, \\ y_0 &= \frac{1}{d+1} \sum_{k=1}^{d+1} f(v_k) - Mdr, \\ h_0 &= Mdr - \frac{1}{d+1} \sum_{k=1}^{d+1} (f(v_k) - m), \\ a_0 &= (c - ru_l, m), \end{aligned}$$

*with*

$$v_k = c - ru_k, \quad k = 1, \ldots, d+1, \quad \textit{the dual vertices of } D,$$

$$m = \min_k \{f(v_k)\},$$

$$l = \textit{an index associated with the lowest evaluation, that is, } f(v_l) = m.$$

We pause here to explain this construction in more detail. After evaluating at $v_k$, for $k = 1, 2, \ldots, d+1$, the initial simplex $T_0$ is constructed by removing $d+1$ "upside down" $(d+1)$-dimensional simplex cones. Each of such cones has apex at $(v_k, f(v_k))$ and its intersection with a hyperplane $y = f(v_k) - tM/d$, for $t > 0$, forms a $d$-dimensional regular simplex with vertices at $v_k - tu_i, i = 1, 2, \ldots, d+1$. The initial simplex $T_0$ is then described by the apex $(x_0, y_0)$, the height $h_0$ and a minimum evaluation point $a_0 = (v_l, f(v_l))$. We use a standard domain $D$ (a hexagon for $d = 2$) because under such a standard domain $D$ the initial simplex forms a true lower bounding function for $f$. Figure 4.5 illustrates this construction for the case $d = 2$. For convenience we have assumed $f(v_1) = f(v_2) = f(v_3)$ in the illustration.

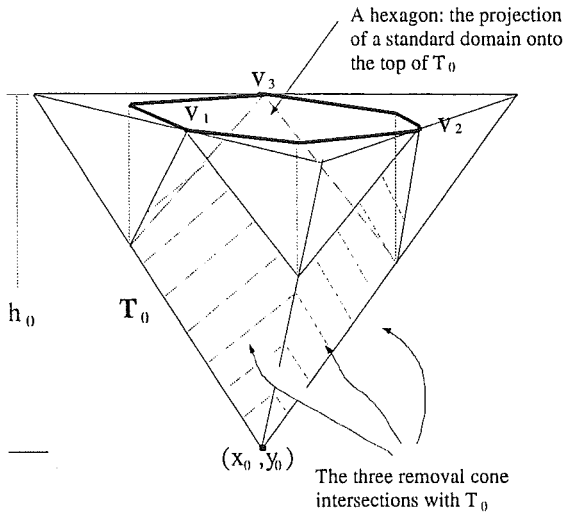



**Figure 4.5** Construction of the initial simplex for the case $d = 2$ and $f(v_1) = f(v_2) = f(v_3)$. Here the hexagon on the top of initial simplex $T_0$ is the projection of a standard domain.

For simplicity, we describe the construction of the initial simplex for the case $d = 2$ only. The construction for the case $d > 2$ can be done similarly.

To calculate $(x_0, y_0)$ and $h_0$ for $d = 2$, we solve the following linear equations which are the intersection planes of the removal cones with $T_0$ :

$$y = f(v_k) - 2Mu_k \cdot (x - v_k) \quad \text{for} \quad k = 1, 2, 3.$$

We consider the case $c = 0$ first. Adding the above three equations and using the fact that $v_k = -ru_k$ and $u_1 + u_2 + u_3 = 0$, we have

$$3y = \sum_{k=1}^{3} f(v_k) - \sum_{k=1}^{3} Mdu_k \cdot (x - v_k) = \sum_{k=1}^{3} f(v_k) - 3Mdr.$$

Dividing both sides by three, we find

$$y_0 = \frac{1}{3} \sum_{k=1}^{3} f(v_k) - Mdr.$$

Then $h_0 = m - y_0 = Mdr - (1/3) \sum_{k=1}^{3} (f(v_k) - m)$, with $m = \min_{1 \le k \le 3} \{f(v_k)\}$.

Since $x_0$ is the centre of the top of the initial simplex, it will be the origin if all function values are equal at the $v_k$. Suppose that $f(v_k)$ are not all equal for $k = 1, 2, 3$. A higher evaluation value at a $v_k$ will force $x_0$ to move in the direction $u_k$. Let $x_0 = \sum_{k=1}^{3} c_k (f(v_k) - m)u_k$, a linear combination of $(f(v_k) - m)u_k$, with coefficients $c_k$. We can choose $c_l$ arbitrarily, as $f(v_l) = m$. Without loss of generality, we may assume that $l = 3$ and $f(v_1), f(v_2) > m$.

The linear equations of the removal planes become

$$\frac{1}{3} \sum_{k=1}^{3} f(v_k) - 2Mr = f(v_k) - 2Mu_k \cdot (\sum_{k=1}^{2} c_k (f(v_k) - m)u_k - v_k) \quad \text{for} \quad k = 1, 2.$$

Notice that $u_k \cdot u_k = 1$ and $u_1 \cdot u_2 = -1/2$, so that the above equations become

$$2M(f(v_1) - m)c_1 - M(f(v_2) - m)c_2 = f(v_1) - \frac{1}{3} \sum_{k=1}^{3} f(v_k)$$

$$-M(f(v_1) - m)c_1 + 2M(f(v_2) - m)c_2 = f(v_2) - \frac{1}{3} \sum_{k=1}^{3} f(v_k)$$

with solution $c_1 = c_2 = 1/(M(d+1))$. We also choose $c_3 = 1/(M(d+1))$. This gives the solution $x_0 = (1/3M) \sum_{k=1}^{3} (f(v_k) - m)u_k$, as required.

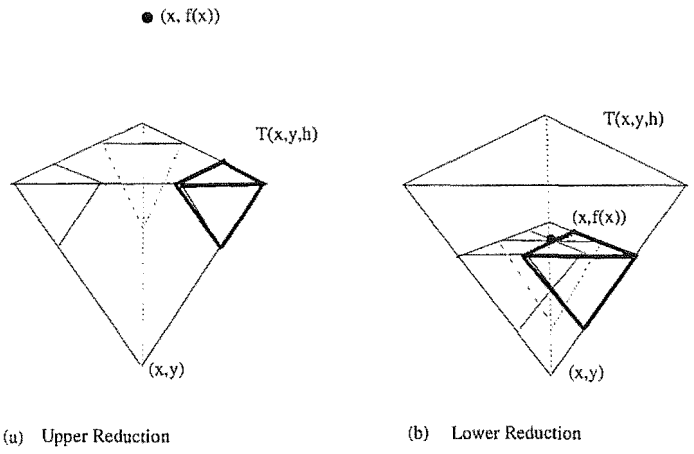
For the case where $c \neq 0$, the centre $x_0$ is translated by $c$, while $y_0$ and $h_0$ will not change. This gives the result

$$\begin{aligned} x_0 &= c + \frac{1}{3M}\sum_{k=1}^{3}(f(v_k) - m)u_k, \\ y_0 &= \frac{1}{3}\sum_{k=1}^{3} f(v_k) - 2Md, \\ h_0 &= 2Mr - \frac{1}{3}\sum_{k=1}^{d+1}(f(v_k) - m). \end{aligned}$$

A straightforward generalisation of the above calculations to the case $d > 2$ leads to the result given in Definition 4.1.2.

### Simplex and system reduction

Given a standard simplex $T = T(x, y, h)$, an evaluation of $f$ at $x$ allows us to remove the interior of $(x, f(x)) - \nabla$ from $T$ and truncate $T$ at height $f(x)$, leaving a region which is again a union of at most $d + 1$ smaller standard simplexes. If $(x, f(x))$ is on or above the top of $T$ it is termed an upper reduction, while if $(x, f(x))$ is below the top of $T$ it is termed a lower reduction. This is illustrated in Figure 4.6.




(a) Upper Reduction (b) Lower Reduction

**Figure 4.6** Simplex reduction when $d = 2$: three small standard simplexes are left when the removal cone is withdrawn from the large standard simplex.

We now derive the expressions for the new simplexes after upper and lower reduction. After an evaluation of $f$ at $x$, of the simplex $T = T(x, y, h)$, we need

to determine the apex $(x'(k), y'(k))$ and height $h'(k)$ of the new simplexes for $k = 1, 2, ..., d+1$. It is easy to see that $x'(k)$ will be located along the direction $x + u_k$. Therefore we consider the intersection of the simplex $T(x, y, h)$ with the vertical plane along $u_k$ and passing through $(x, y)$. Figure 4.7 (A) illustrates the case for $d = 2$. For simplicity, we denote $(x'(k), y'(k))$ as $(x', y')$, $h'(k)$ as $h'$ and use $r = h/M$, the radius of the top of $T(x, y, h)$.

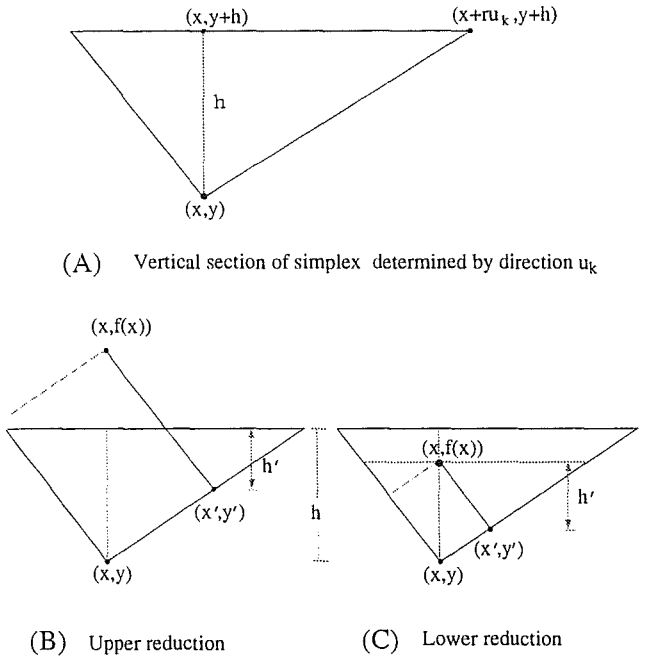



**Figure 4.7** The action of simplex reduction along $u_k$ : the section before reduction is shown in (A) with base point $(x, y)$ and height $h$ while upper reduction is shown in (B) and lower reduction in (C). The new simplex has base point $(x', y')$ and height $h'$.

The equation of the "segment" from point $(x, y)$ to point $(x + ru_k, y + h)$ is

$$X - x = \lambda u_k, \qquad Y - y = M||X - x||, \quad 0 \leq \lambda \leq r$$

The equation of the $k$th removal hyperplane, with normal $(-u_k, -\frac{1}{dM})$, of removal cone $(x, f(x)) - \nabla$ is

$$Y - f(x) = -dMu_k \cdot (X - x).$$

The base point $(x', y')$ of the new simplex along direction $u_k$ from $x$ is then determined by solving the simultaneous equations formed by the above segment and hyperplane. This leads to

$$f(x) - dMu_k \cdot (X - x) = y + M||X - x||,$$

$$f(x) - dM\lambda = y + M\lambda, \quad \text{then}$$

$$\lambda = \frac{f(x) - y}{(d+1)M} \quad \text{therefore}$$

$$x' = x + \frac{f(x) - y}{(d+1)M}u_k, \qquad y' = y + \frac{f(x) - y}{d+1}.$$

The height $h'$ is then determined as follows:

For upper reduction,

$$h' = y + h - y' = h - \frac{1}{d+1}(f(x) - y).$$

For lower reduction,

$$h' = f(x) - y' = \frac{d}{d+1}(f(x) - y).$$

This is illustrated in Figure 4.7 (B) and (C).

Let $\mathcal{T}$ denote the set of simplexes comprising the reduction of $T$. The above simplex reduction is formally described in the following definition.

**Definition 4.1.3** *Let $T(x, y, h)$ be a standard simplex. Two cases occur:*

1. Upper reduction (*when* $h \leq f(x) - y$)

*If* $h \leq f(x) - y \leq (d+1)h$, *then the reduction of* $T$ *is*

$$\mathcal{T} = \left\{ T[x + \frac{f(x) - y}{M(d+1)}u_i, y + \frac{f(x) - y}{d+1}, h - \frac{1}{d+1}(f(x) - y)] : i = 1, \ldots, d+1 \right\},$$

*else if* $(d+1)h < f(x) - y$, *then* $\mathcal{T}$ *is the empty set.*

2. Lower reduction (*when* $f(x) - y < h$)

*If* $0 \leq f(x) - y < h$, *then the reduction of* $T$ *is*

$$\mathcal{T} = \left\{ T[x + \frac{f(x) - y}{M(d+1)}u_i, y - \frac{y - f(x)}{d+1}, \frac{d}{d+1}(f(x) - y)] : i = 1, \ldots, d+1 \right\},$$

*else if* $f(x) - y < 0$, *then* $\mathcal{T}$ *is the empty set.*

In the procedure of simplex reduction, no global minima are removed as the simplex-based removal cone $(x, f(x)) - \nabla$ is contained in the spherically-based removal cone $(x, f(x)) - \mathcal{O}$, which in turn does not remove any global minima of the objective function $f \in L(M)$.

In *system reduction* some of the simplexes are reduced in the current system. In the following definition, $J$ indexes all simplexes in the current system, and $I$ indexes the simplexes to be reduced.

**Definition 4.1.4** (System reduction) *Let $\mathcal{S} = (\mathcal{T}, a)$ be a uniform system inside the initial simplex $T_0$, where $\mathcal{T} = \{T_j\}_{j \in J}$, and let $I$ be a non-empty subset of $J$. A reduction of $\mathcal{S}$ is a system $\mathcal{R}(\mathcal{S}) = (\mathcal{T}', a')$ where*

$$\mathcal{T}' = \bigcup_{j \in I} \mathcal{T}_j \bigcup_{j \in J \setminus I} \{T_j\}, \quad \text{where } \mathcal{T}_j \text{ is the reduction of simplex } T_j,$$

$$a' = \begin{cases} a, & \text{if no lower reductions occur, else} \\ (x, f(x)) & \text{such that } f(x) = \min_{j \in I}\{f(x_j)\}. \end{cases}$$

The process of system reduction does not remove any global minima of $f$ since each simplex reduction procedure does not do so.

### System elimination

Following a system reduction, parts of some simplexes may lie above the lowest function value recorded. The system is tightened up in the following way:

**Definition 4.1.5** (System elimination) *The eliminated system associated with the system $\mathcal{S} = (\mathcal{T}, a)$ inside $T_0$ is $\mathcal{E}(\mathcal{S}) = (\mathcal{T}', a)$ where*

$$\mathcal{T}' = \{T \cap P^- \ : \ T \in \mathcal{T}\},$$

*with $P^-$ the closed half-space of $\boldsymbol{R}^{d+1}$ below $a$.*

Again, no global minima of $f$ are removed during this process. This is due to Fact 2 described for a Lipschitz continuous function $f \in L(M)$, in Section 2.1.

The natural domain $D$ is contained in the projection of the top of the initial simplex. The lower boundary of the initial simplex provides a lower bound for $f$ over $D$, not over the top of $T_0$. As the above procedures are performed on the initial simplex onwards, the tops of some subsimplexes may locate or partly locate out of $D$. In order to ensure that the algorithm converges to the global minimum of $f$ over the domain $D$, the following assumption is needed:

**Assumption 4.1.1** *The global minimum of $f$, over the projection of the top of $T_0$ onto $\boldsymbol{R}^d$, occurs in $D$.*

If we denote the system at the $k$th iteration by $\mathcal{S}_k$, then we can summarize the multidimensional bisection algorithm as:

**Multidimensional Bisection Algorithm**

**Initial step:** Form $\mathcal{S}_0$, the initial system.

**Iterative step:** Let $\mathcal{S}_{k+1} = \mathcal{E}(\mathcal{R}(\mathcal{S}_k))$. Repeat until a stopping criterion is satisfied.

For the case where $I = J$, the multidimensional bisection algorithm reduces all simplexes obtained from the previous step within one iteration. We term this strategy the Full Multidimensional Bisection Algorithm (FMBA). Of particular interest is the case where $I$ picks out the deepest simplex in the system at each stage, so $|I| = 1$. We term this strategy the Deepest Multidimensional Bisection Algorithm (DMBA).

### Convergence of the multidimensional bisection algorithm

The convergence of the variation of the system sequence is guaranteed by the next lemma.

**Lemma 4.1.1** *Consider FMBA and let $\mathcal{S}$ be the uniform system formed at some iteration. Then*

$$V(\mathcal{E}(\mathcal{R}(\mathcal{S}))) \leq \frac{d}{d+1} V(\mathcal{S}).$$

Proof: Let $\mathcal{S} = (\mathcal{T}, a)$, and select the $T_j$ in $\mathcal{T}$ whose reduction contains a point at least as low as any point in any other reduction. Letting $V(\mathcal{T}_j)$ be the variation of $\mathcal{T}_j$, we have

$$V(\mathcal{E}(\mathcal{R}(\mathcal{S}))) \leq V(\mathcal{T}_j) \leq \frac{d}{d+1} h_j \leq \frac{d}{d+1} V(\mathcal{S}).$$

□

We use the following notation for describing the convergence results. After the multidimensional bisection algorithm completes iteration $k$, there is a system $\mathcal{S}_k$ which consists of a set of simplexes $\mathcal{T}_j$ and associated $a_k$. Let $\alpha_k$ be the value of the smallest function evaluation to date and $\beta_k$ be the level of the lowest apex in the system of simplexes.

We need the following lemma for the convergence result for DMBA. The proof of this lemma can be found in [78].

**Lemma 4.1.2** *Let $\mathcal{R}$ be the reduction process applied in DMBA. Assume that the algorithm has completed iteration $k_0$. Let $\alpha_{k_0}$ be the smallest function value evaluated to date and $\beta_{k_0}$ be the level of the lowest value of the base points of simplexes in the current system. Then there exists an integer $K$ such that*

$$\alpha_K - \beta_K = V(\mathcal{S}_K) \leq \frac{d}{d+1} V(\mathcal{S}_{k_0}) = \frac{d}{d+1}(\alpha_{k_0} - \beta_{k_0}).$$

The idea behind the proof is the following. Reducing the system variation $V(\mathcal{S}_{k_0})$ by a factor of $d/(d+1)$ can be performed by reducing all simplexes which penetrate beneath the level $\alpha_{k_0} - (d/(d+1))V(\mathcal{S}_{k_0})$, a distance $(1/(d+1))V(\mathcal{S}_{k_0})$ above $\beta_{k_0}$, the bottom of the current system. An upper reduction of such a simplex would suit this aim, while a lower reduction of such a simplex could well introduce $d+1$ new simplexes below the level $\alpha_{k_0} - (d/(d+1))V(\mathcal{S}_{k_0})$. Each such lower reduction, however, lowers $\alpha_{k_0}$, the current removal level. All the new simplexes lie above the bottom $\beta_{k_0}$, so preventing a build up of simplexes which need to be reduced. Eventually all simplexes beneath the current level $\alpha_{k_0} - (d/(d+1))V(\mathcal{S}_{k_0})$ are reduced by a factor $(1/(d+1))V(\mathcal{S}_{k_0})$.

In [78], two conditions $C_1$ and $C_2$ were introduced. Condition $C_1$ requires that all deepest simplexes in the system at the end of each iteration are eventually reduced and Condition $C_2$ insists that all simplexes in the system at the end of each iteration are eventually reduced. It was proved in [78], that Condition $C_1$ ensures the system variation converges to zero while condition $C_2$ also ensures localisation convergence. It easy to see that DMBA is an algorithm satisfying $C_1$ while FMBA is an algorithm satisfying $C_2$. We present the convergence results for DMBA and FMBA in the following theorems.

Let $f_*$ be the global minimum of the objective function over a natural domain $D$. Consider DMBA and FMBA for solving problems $P$ and $Q$. We have

**Theorem 4.1.1** *For DMBA* $(I = 1)$ *the following hold:*

*i)* $\alpha_k \downarrow f_*$ *and* $\beta_k \uparrow f_*$ *as* $k \to \infty$,

*ii)* $\lim_{k\to\infty} f(x_k) = f_*$,

*iii)* $A \subseteq E \subseteq L_\infty$,

*iv)* $\lim_{k\to\infty} \inf_{x\in E} ||x - x_k|| = 0$.

**Theorem 4.1.2** *For FMBA* $(I = J)$ *the following hold:*

*i)* $\alpha_k \downarrow f_*$ *and* $\beta_k \uparrow f_*$ *as* $k \to \infty$,

*ii)* $\lim_{k\to\infty} f(x_k) = f_*$,

*iii)* $A = E = L_\infty$,

*iv)* $\lim_{k\to\infty} \inf_{x\in E} ||x - x_k|| = 0$.

The proofs of Theorems 4.1.1 and 4.1.2 follow immediately from Theorems 4.1 and 4.2 of [78]. The proof of Theorem 4.1 (ii) in [78] (Theorem 4.1.1 above) is incomplete, so we fill in the gap here.

Suppose $f(x_k)$ does not tend to $f_*$. Since $\alpha_k - \beta_k$ decreases to 0 and $\alpha_k \leq f_* \leq \beta_k$, for $\epsilon > 0$ there must exist a subsequence $x_{j_k}$ such that $f(x_{k_j}) > f_* + \epsilon$. By compactness of $D$ there exists a further subsequence $x_{j_{k_l}}$ say which converges to some $z$ in $D$. But for sufficiently large $l$, an upper reduction at $x_{j_{k_l}}$ will remove the bracket in a region of $(z, f(z))$, a contradiction.

Notice that, for the deepest strategy of multidimensional bisection, only one of the simplexes with the deepest base point is selected to be reduced at each iteration, so some of the simplexes in the current system may be left without reduction for ever. This leads to the weaker convergence result (iii) of Theorem 4.1.1. To obtain the relation $L_\infty \subseteq A$, further modification to DMBA is needed. In fact result (iii) of Theorem 4.1.1 cannot be improved without further modification to the deepest strategy DMBA. An example will be given in Section 4.6 for which both inclusion relations in (iii) of Theorem 4.1.1 are strict.

## 4.2 Acceleration of multidimensional bisection

The character of the algorithm depends upon the choice of simplexes reduced in an iteration. For the case that $I = J$, all simplexes are reduced, so ensuring that the variation of the system is reduced by at least a factor of $d/(d+1)$ at each

full iteration. Theorem 4.1.1 shows that deepest point reduction is sufficient to ensure that the variation converges to zero. This choice creates the multidimensional bisection analogue of the strategies employed by Piyavskii-Shubert and Mladineo.

No matter what reduction strategy we employ, there are two immediate failings of the algorithm:

(i) For $d$ greater than one, an evaluation over one simplex frequently generates a removal cone which is capable of removing material from neighbouring simplexes. The algorithm, as described so far, does not effect this action, which we term complete reduction.

(ii) In reality, we can remove a spherically based cone at an evaluation point. The simplex based cone merely approximates this cone, and the approximation worsens as $d$ increases. A method is needed, which we term spherical reduction, which retains the simplicity of simplexes, yet utilises the power of the spherical removal.

We now describe the complete reduction, $A^c$, and the spherical reduction, $A^s$, acceleration procedures.

### Complete reduction

The *complete reduction* algorithm $A^c$ recognizes that the notion of simplex reduction described in the previous section can be generalised. So far, reduction is done to a simplex when the evaluation occurs over its apex (or deepest point). This restriction is not necessary. Let $z_0 = (x_0, f(x_0))$ be any point on the epigraph of $f$ in $\mathbf{R}^{d+1}$. Given a standard simplex $T$, $z_0 - \nabla$ can be used as a removal cone resulting in $T \backslash (z_0 - \nabla)$ being a union of at most $d+1$ standard simplexes. Figure 4.8 illustrates this statement. Note that standard simplexes of unequal size are left after such a reduction.

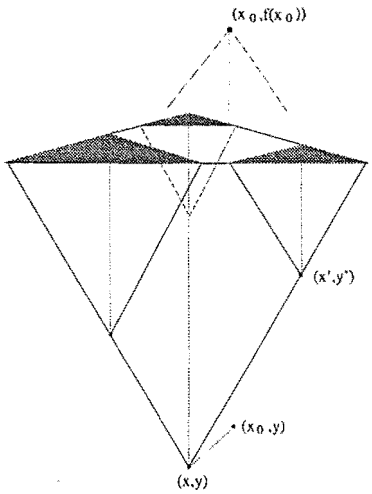



**Figure 4.8** Simplex reduction when the evaluation is not over the apex. Note the remaining simplexes are standard, but not of equal size.

The determination of the remaining new non-even simplexes can be easily done by modifying the calculation employed in finding the form of the new even simplexes in Section 4.1. In fact, if we evaluate $f$ at $x_0$, with $x_0$ not necessarily the base point of the simplex $T = T(x, y, h)$, we will have the following.

The base point $(x', y')$, of the new simplex along direction $u_k$ from $x$ can be obtained in a similar way to that described in Section 4.1 (for the case of evaluation at the base point $(x, y)$). That is, point $(x', y')$ is the solution of the simultaneous equations:

$$X - x = \lambda u_k, \qquad Y - y = M||X - x||, \quad 0 \leq \lambda \leq r, \quad \text{and}$$

$$Y - f(x_0) = -dMu_k \cdot (X - x_0).$$

This leads to

$$f(x_0) - dMu_k \cdot (X - x_0) = y + M||X - x||,$$

$$f(x_0) - dMu_k \cdot (X - x + x - x_0) = y + M\lambda, \qquad \text{whence we have}$$

$$\lambda = \frac{f(x_0) - y}{(d+1)M} - \frac{d}{d+1} u_k \cdot (x - x_0).$$

Therefore the base points of the new irregular simplexes are

$$x' = x + [\ \frac{f(x_0) - y}{(d+1)M} - \frac{d}{d+1} u_k \cdot (x - x_0)\ ]u_k,$$

$$y' = y + \frac{f(x_0) - y}{d+1} - \frac{dM}{d+1} u_k \cdot (x - x_0),$$

for $k = 1, 2, \ldots, d+1$. The height of the new simplexes is

$$h'(k) = \begin{cases} (y+h) - y' & \text{for upper reduction or} \quad f(x_0) > y + h, \\ f(x_0) - y' & \text{for lower reduction or} \quad f(x_0) \leq y + h. \end{cases}$$

The above results enable us to remove from more simplexes when we evaluate at just one point. This can be done by noting that a single evaluation for a given simplex at its base point is in fact a non-base point evaluation for any other simplex. When we incorporate this process into the algorithm we term it "complete reduction".

### Spherical reduction

The *spherical reduction* algorithm $A^s$ removes more material than the raw multidimensional bisection algorithm. Pictured in Figure 4.9 is the triangular top $T'$ of $T(x, y, h)$ and the cross-section $D'$ of the removal cone $(x, f(x)) - \nabla$ through the plane of the simplex top. We are really permitted to remove $(x, f(x)) - \mathcal{O}$, a much larger volume, whose cross-section is shown as $S$ in the diagram. It is now clear that we can remove a simplex based cone at an effective evaluation point higher than $(x, f(x))$. Its cross-section through the plane of $T'$ is shown as $D_s$, evidently the largest simplex dually oriented to the top of $T(x, y, h)$ whose intersection with $T'$ is contained in $S$.

The key to spherical reduction is the acceleration function $A$ which relates the radius of $S$ to the radius of $D_s$, as shown in Figure 4.9. We standardise by taking $T'$ in Figure 4.9 to have unit radius, whence $A$ will be a function from [0,1]

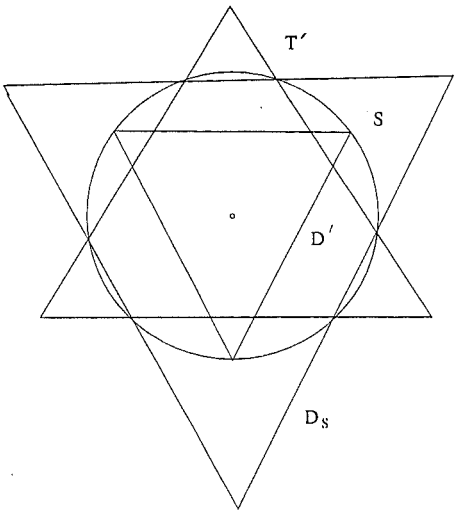



**Figure 4.9** The basis of spherical reduction: if $T'$ were the top of a simplex in the system, and $S$ the cross-section of the spherical removal cone, then we could effectively remove a simplex based cone with cross-section $D_s$ from the simplex with top $T'$.

to [0,$d$]. The following theorem describes $A$. The proof of the following theorem was given by Wood in [78].

**Theorem 4.2.1** *Let $T'$ be a regular $d$-simplex of unit radius and $S$ be a $d$-sphere with the same centre, and radius $r$, $0 \leq r \leq 1$. Then the radius, $A(r)$, of the largest regular $d$-simplex, $D_s$, dually oriented to $T'$, sharing the common centre and such that $T' \cap D_s \subseteq S$, is given by the piecewise formula, with $d$ parts:*

$$A(r) = A(r,i) \quad for \quad \sin\theta_{d,i} \leq r \leq \sin\theta_{d,i-1}, \quad i = d, d-1, \ldots, 1.$$

*where the $A(r,i)$ are defined recursively as:*

$$A(r,d) = r \quad for \quad 0 \leq r \leq \sin\theta_{d,d-1} \quad and$$
$$A(r,i) = A(\sin\theta_{d,i}, i+1) + \frac{\sqrt{r^2 - \sin^2\theta_{d,i}}}{\prod_{j=i+1}^{d} \tan\theta_{j,1}}$$

*for $\sin\theta_{d,i} \leq r \leq \sin\theta_{d,i-1}$ and $i = d-1, d-2, \ldots, 1$, where $\theta_{j,k}$ is the angle between the line joining vertex and centroid, and the line joining vertex to centroid of a $k$-dimensional face, in a $j$-simplex, for $j \geq k$. We define $\theta_{d,0} = \pi/2$.*

In [78], the following readily used formula for calculating $A(r,i)$ is given:

$$\sin\theta_{d,i} = \sqrt{\frac{d-i}{d(i+1)}} \quad \text{and} \quad \tan\theta_{j,1} = \sqrt{\frac{j-1}{j+1}} \quad \text{for all } 1 \le i,j \le d.$$

Table 4.1 shows the acceleration function $A$ for the case $d = 2$ and 3.

| $d=2$ | | $d=3$ | |
|---|---|---|---|
| $r$ | $A(r)$ | $r$ | $A(r)$ |
| $[0, 1/2]$ | $r$ | $[0, 1/3]$ | $r$ |
| $[1/2, 1]$ | $1/2 + \sqrt{3(r^2 - (1/2)^2)}$ | $[1/3, 1/\sqrt{3}]$ | $1/3 + \sqrt{2(r^2 - (1/3)^2)}$ |
| | | $[1/\sqrt{3}, 1]$ | $1 + \sqrt{6(r^2 - (1/\sqrt{3})^2)}$ |

Table 4.1 The acceleration function, $A(r)$, for spherical reduction, for $d = 2$ and 3.

Once we know the function $A$, the spherical acceleration algorithm $A^c$ is easy to implement. This can be done by lifting the function evaluation at $x$, for simplex $T(x,y,h)$, to an "effective" height $hA\left[\frac{f(x)-(y+h)}{h}\right] + (y+h)$, which ensures the simplicial removal cone intersects the top of $T(x,y,h)$ in $D_s$. This allows us to remove more from $T$ for a single upper reduction. The accelerated spherical upper reduction procedure is then given in the following definition. This replaces (1) in Definition 4.1.3.

**Definition 4.2.1** *Let $T(x,y,h)$ be a standard simplex. If $h \le f(x) - y \le 2h$, then the upper reduction of this simplex is*

$$\mathcal{T} = \left\{ T[x + \frac{F(x)-y}{M(d+1)}u_i, y + \frac{F(x)-y}{d+1}, h - \frac{1}{d+1}(F(x)-y)] : \ i = 1,\ldots,d+1 \right\},$$

*else if $2h < f(x) - y$, then $\mathcal{T}$ is the empty set.*

*Here $F(x) = hA\left[\frac{f(x)-(y+h)}{h}\right] + (y+h)$ is the effective evaluation of $f$ at $x$.*

### Complete spherical reduction

Spherical acceleration utilises the power of the spherically based removal cone, but only for removal from the simplex over which the evaluation was made. This acceleration can be combined with complete reduction to extend to all simplexes which intersect with the spherically based removal cone. We term this *complete spherical reduction* $A^{cs}$, a technique we now describe. An evaluation over a point $x$ determines a fixed spherically based removal cone. Consider a simplex in the system which meets this removal cone. Denote the top of this simplex, shaded in Figure 4.10, by $T$. Construct a dummy standard simplex, with top $T'$, the smallest centred on $x$ and containing $T$. Spherical reduction on this dummy simplex would generate a simplex based removal cone at an effective evaluation point higher than $(x, f(x))$. Its cross-section at the level of the tops is shown as $D'$ in Figure 4.10. Now remove this cone from the original simplex, as in the complete reduction procedure. This combined process allows us to remove more than would spherical or complete reduction alone. Figure 4.10 illustrates this case.

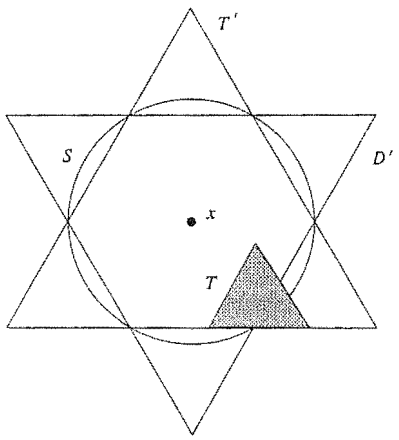



**Figure 4.10** The basis of complete spherical reduction.

## 4.3 Numerical results

An implementation of the deepest point algorithm has been written in `matlab` by W.P. Baritompa. This runs the raw "deepest point" algorithm ($A$), and either or both of the two acceleration schemes. The `matlab` implementation of the algorithm is a "dual" implementation, in which each simplex is held by means of the dual coordinates of the sloping facets of the simplex. The dual coordinate of a facet is the inner product of any vector from the origin to a point on the facet with a unit vector orthogonal to the facet.

We first report on the relative performance of the four schemes, $A, A^c, A^s$ and $A^{cs}$, using three test functions which are already in the literature, and two members of a family of test functions suggested by recent work of Mladineo in [44]. The Mladineo functions have the appearance of upside-down mountain ranges, being non-differentiable at the global minimum.

For each of the test functions, we present the function $f$, the location of the global minima $x_*$, the Lipschitz constant $M$ which we adopted, the initial feasible domain $D$ (in terms of the centre $c$ and radius $r$, see Definition 4.1.1 (5)) and the variation of the initial simplex, $V_0$.

Different runs were created by varying the location of the first evaluation following formation of the initial simplex. Thereafter the deepest point algorithm was used. Each run is terminated after 100 function evaluations. For each function we report the average, over a number of runs, of

i) the location of the least evaluation, and

ii) the ratio of the final variation to the initial variation.

The deviation from the true global minimum is available by comparing the final component of the located point with the final component of the true point in the following tables.

### Test function 1. Goldstein and Price (GOLDPR)

$$f(x_1', x_2') = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \cdot \left[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right] / \delta$$

where $x_1 = 4x_1' - 2$, $x_2 = 4x_2' - 2$ and $\delta = 1,015,000$. We take $M = 50$; $c = (0.5, 0.5)$, $r = 0.7098$, then $V_0 = 70.3$. Our initial domain cuts the corners from $[0,1] \times [0,1]$ in order to avoid a second minimum. The true minimum of $f$ is $(x_*, f(x_*)) = (0.5000, 0.2500, 0.0000)$.

| | Algorithm minimum | Rel. varn | No. runs |
|---|---|---|---|
| $A$ | (0.5014, 0.2565, 0.0000) | 0.1788 | 10 |
| $A^c$ | (0.4681, 0.2681, 0.0000) | 0.1475 | 10 |
| $A^s$ | (0.5014, 0.2565, 0.0000) | 0.1788 | 10 |
| $A^{cs}$ | (0.4687, 0.2681, 0.0000) | 0.1272 | 10 |

Table **4.2** Multidimensional bisection performance on test function GOLDPR.

### Test function 2. Branin (RCOS)

$$f(x_1', x_2') = \left[a(x_2 - bx_1^2 + cx_1 - d)^2 + e(1 - f)\cos x_1 + e\right] / \lambda$$

where $a = 1$, $b = 5.1/(4\pi^2)$, $c = 5/\pi$, $d = 6$, $e = 10$, $f = 1/(8\pi)$, $\lambda = 308.1$ and $x_1 = 15x_1' - 5$, $x_2 = 15x_2'$. $M = 10$ ; $c = (0.5, 0.5)$, $r = 0.7887$ ; $V_0 = 15.63$. Here $D$ is the smallest hexagon containing $[0,1] \times [0,1]$.

There are three global minimisers, at $(0.5428, 0.1517)$ , $(0.1239, 0.8183)$ and $(0.9617, 0.1650)$ with minimum $f_* = 0.0013$.

| | Algorithm minimum | Rel. varn | No. runs |
|---|---|---|---|
| $A$ | (0.5436, 0.1618, 0.0035) | 0.1690 | 7 |
| $A^c$ | (0.5281, 0.1553, 0.0032) | 0.1240 | 4 |
| $A^s$ | (0.5436, 0.1618, 0.0035) | 0.1731 | 7 |
| $A^{cs}$ | (0.5281, 0.1553, 0.0032) | 0.1242 | 4 |
| $A$ | (0.1308, 0.7840, 0.0027) | 0.1695 | 3 |
| $A^c$ | (0.1361, 0.7558, 0.0029) | 0.1259 | 5 |
| $A^s$ | (0.1308, 0.7831, 0.0027) | 0.1695 | 4 |
| $A^{cs}$ | (0.1361, 0.7558, 0.0029) | 0.1242 | 5 |
| $A$ | (0.9423, 0.1446, 0.0026) | 0.1644 | 1 |
| $A^c$ | (0.9665, 0.1805, 0.0035) | 0.1241 | 2 |
| $A^s$ | (0.9423, 0.1446, 0.0026) | 0.1644 | 1 |
| $A^{cs}$ | (0.9665, 0.1804, 0.0035) | 0.1231 | 2 |

Table **4.3** Multidimensional bisection performance on test function RCOS.

### Test function 3. Mladineo (FUNCT2)

$$f(x_1, x_2) = -\left[\sin(4x_1 + 1) + 2\sin(6x_2 + 2)\right]$$

For the hexagonal initial domain we use the smallest hexagon containing $[0,1]\times[0,1]$. Also

$M = 12.65$; $c = (0.5, 0.5)$, $r = 0.7887$; $V_0 = 19.24$. There are two global minimisers at $(0.1427, 0.9759)$ and $(0.1427, -0.0715)$ with global minimum $f_* = -3.0000$.

| | Algorithm minimum | Rel. varn | No. runs |
|---|---|---|---|
| $A$ | (0.1249, 0.9629, −2.9916) | 0.0855 | 10 |
| $A^c$ | (0.1086, 0.9737, −2.9909) | 0.0654 | 8 |
| $A^s$ | (0.1285, 0.9745, −2.9927) | 0.0790 | 10 |
| $A^{cs}$ | (0.1286, 0.9699, −2.9940) | 0.0588 | 8 |

Table 4.4 Multidimensional bisection performance on test function FUNCT2.

### Test function 4. MLADINEO (d,m)

We now define a class of functions on which the type of algorithm we are using thrives. They have the appearance of down-under mountain ranges, and were sug-

gested by recent work of Mladineo in [44]. For $k = 1, \ldots, m$, let $M_k$ be a positive real number, and $c_k$ be a point in $\mathbf{R}^d$. Define

$$f(x) = \min\{-M_k \exp(- \parallel x - c_k \parallel) : k = 1, \ldots, m\} \quad \text{on } \mathbf{R}^d.$$

Then $x_* = c_{k_0}$; $f(x_*) = -M_{k_0}$, where $k_0$ is such that $M_{k_0} = \max\{M_k : k = 1, \ldots, m\}$.

**Mladineo (2,3)** Two variables and three inverted peaks.

We choose $d = 2$, $m = 3$ and let $c_1$, $c_2$ and $c_3$ be $(-0.5, -0.5)$, $(0.6, -0.4)$ and $(0, 0.8)$ respectively, with $M_k = \sqrt{k}$, for $k = 1, 2, 3$. We take $M = \sqrt{3}$, $c = (0, 0)$, $r = 1$, then $V_0 = 3.435$; $x_* = (0, 0.8000)$ with $f_* = -1.7321$.

**Mladineo (4,3)** Four variables and three inverted peaks.

We choose $d = 4$, $m = 3$ and let $c_1$, $c_2$ and $c_3$ be $.7u_1 + .5u_2 + .6u_3 + .8u_4$, $-.6u_1 - .7u_2 - .8u_3 + .5u_4$ and $.8u_5$ respectively, with $M_k = \sqrt{k}$, for $k = 1, 2, 3$. Here $u_1, \ldots, u_5$ are the directions in $\mathbf{R}^4$ defining the vertices of the simplex top. We take $M = \sqrt{3}$, $c = (0, 0, 0, 0)$, $r = 1$, then $V_0 = 6.897$; $x_* = (0, 0, 0, 0.8000)$ with $f_* = -1.7321$.

| | Algorithm minimum | Rel. varn | No. runs |
|---|---|---|---|
| | **MLADINEO (2,3)** | | |
| $A$ | (0.0002, 0.8010, −1.7284) | 0.0068 | 10 |
| $A^c$ | (0.0000, 0.8001, −1.7317) | 0.0004 | 10 |
| $A^s$ | (0.0000, 0.8005, −1.7296) | 0.0025 | 10 |
| $A^{cs}$ | (0.0000, 0.8000, −1.7320) | 0.0000 | 10 |
| | **MLADINEO (4,3)** | | |
| $A$ | (0.01, −0.01, 0.00, 0.73, −1.58) | 0.3755 | 13 |
| $A^c$ | (0.01, −0.01, 0.00, 0.75, −1.60) | 0.2997 | 13 |
| $A^s$ | (0.01, −0.01, 0.00, 0.73, −1.58) | 0.3680 | 13 |
| $A^{cs}$ | (0.01, −0.01, 0.00, 0.75, −1.60) | 0.2696 | 13 |

Table **4.5** Multidimensional bisection performance on test functions MLADINEO (2,3) and MLADINEO (4,3).

For each test function and algorithm type the table shows the average over several runs of the computed minimum and relative variation (the ratio of the final variation to the initial variation). The differing number of runs within RCOS is due to the presence of more than one global minimum; we selected only the runs which converged to the specified global minimum.

Remarks:

i) The algorithms work best for functions where the minima are well-defined, such as FUNCT2 and Mladineo (2,3) and (4,3). The effectiveness of simplex reduction in such cases causes the system variation to reduce rapidly.

ii) For functions which are relatively flat over much of a neighbourhood around the global minimum the variation is slow to reduce. Thus while we may find a good solution early, it takes a lot of later evaluations to confirm that it is successful. GOLDPR and RCOS exhibit this behaviour. This phenomenon described is quite typical in global optimisation.

iii) The algorithms are converging to a global minimum, but it is evident by examining absolute error in Table 4.2 that they are more successful at finding the value of the function at the global minimum (the final coordinate) than the location of the global minimum (the first $d$ coordinates).

iv) Complete reduction produces roughly a 25% reduction in variation, though substantially more when the global minimum is sharply defined, as in Mladineo (2,3). Spherical reduction makes almost no difference for functions which are flat around the global minimum, but does offer an improvement for FUNCT2 and Mladineo (4,3), and a marked one for Mladineo (2,3). Recall that spherical reduction comes into its own only when evaluations lie well above the simplex top. Note that $A^{cs}$ is best overall.

v) Accelerated methods require fewer function evaluations to reach a given variation, but the overheads per function evaluation are higher. For our current

implementation the overall overheads to reach a given accuracy, measured in floating point operations, do not change very much from $A$ to $A^{cs}$.

It is planned in the future to investigate certain higher dimensional test problems.

In [28] and [33] Hansen and Jaumard compared the performance of certain Lipschitz optimisation algorithms as they found the global $\epsilon$-optimal value of 20 univariate and 13 bivariate Lipschitz test functions. Multidimensional bisection is among the algorithms studied. For the bivariate test functions, it showed that the multidimensional bisection requires a large number of function evaluations for most of the test functions, for the given tolerance $\epsilon$, but that the computation time needed is relatively short. The large number of function evaluations is mainly due to the overlapping of the simplexes in a system, while the fast computation is due to the simplicity of finding the lowest base point of the simplexes in the system. They use DMBA for implementing the Wood algorithm and none of the previously described acceleration strategies for multidimensional bisection algorithm were used. Employing such acceleration strategies in the multidimensional bisection algorithm will significantly reduce the number of function evaluations required for $\epsilon$-convergence. Table 4.6 lists six of the bivariate test functions used in [28], with their domains.

| No. | Lipschitz function | Domain |
|---|---|---|
| 1 | $-4xy\sin(4\pi y)$ | $[0,1]\times[0,1]$ |
| 2 | $-\sin(2x+1)-2\sin(3y+2)$ | $[0,1]\times[0,1]$ |
| 4 | $\max(\sqrt{3}x+y,-2y,-\sqrt{3}x)$ | $[-1,1]\times[-1,1]$ |
| 5 | $\lvert y\rvert+e^{-x^2}$ | $[0,10]\times[0,10]$ |
| 8 | $(x-2y-7)^2+(2x+y-5)^2$ | $[-2.5,3.5]\times[-1.5,4.5]$ |
| 10 | $\sin(x+y)+(x-y)^2-1.5x+2.5y+1$ | $[-1.5,4]\times[-3,3]$ |

Table **4.6** The six Hansen-Jaumard test functions with their associated domains.

Table 4.7 compares the number of function evaluations required by $A^c$, $A^s$ and $A^{cs}$ for these six test functions using the same Lipschitz constant $M$ and tolerance $\epsilon$ given on pages 78 and 80 of [28]. The centre $c$ and radius $r$ of the standard domain $D$ which includes the corresponding rectangular domain is also listed.

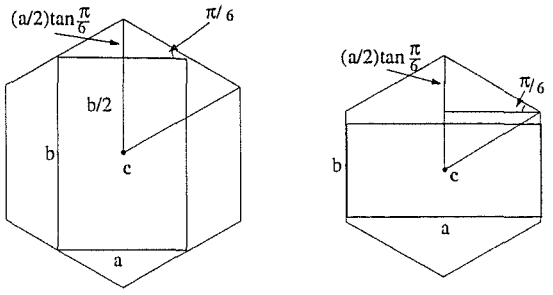
| No. | Tolerance $\epsilon$ | Standard Domain | $M$ | No. of function evaluations | | | |
|---|---|---|---|---|---|---|---|
| | | | | $A$ | $A^c$ | $A^s$ | $A^{cs}$ |
| 1 | 0.355 | $\mathbf{c}=(.5,.5)$, $r=0.7887$ | 50.267 | 3454 | 1064 | 1019 | 691 |
| 2 | 0.0446 | $\mathbf{c}=(.5,.5), r=0.7887$ | 6.318 | 2212 | 1285 | 1902 | 1107 |
| 4 | 0.0141 | $\mathbf{c}=(0,0), r=1.5774$ | 2 | 134 | 82 | 33 | 31 |
| 5 | 0.1 | $\mathbf{c}=(5,5), r=7.8868$ | $\sqrt{2}$ | 391 | 263 | 256 | 175 |
| 8 | 3.66 | $\mathbf{c}=(.5,1.5), r=4.7321$ | 86.313 | 1717 | 1185 | 1393 | 995 |
| 10 | 0.691 | $\mathbf{c}=(.75,0), r=4.5877$ | 17.034 | 1546 | 1035 | 1069 | 752 |

Table 4.7 Comparison of the number of function evaluations required by $A$, $A^c$, $A^s$ and $A^{cs}$ for the six Hansen-Jaumard test functions, over a standard domain with centre $\mathbf{c}$ and radius $r$.

Note the lower number of function evaluations required as the acceleration methods are brought into play.

Hansen and Jaumard in [28] pointed out that it is difficult to determine the standard domain $D$ with the smallest radius $r$ containing a hyperrectangle. We show now how to do this for the case $d = 2$. Figure 4.11 illustrates this case. We used the following formulae to obtain the smallest standard domain $D$ which contains the rectangle specified for each of the above six test functions.

For a given rectangular domain with length $a$, width $b$ and centred at $(c_1, c_2)$, the standard domain $D$ with smallest radius $r$ containing such a rectangle can be determined using elementary geometry:




A) $b \geq a\tan\frac{\pi}{6}$ B) $b < a\tan\frac{\pi}{6}$

**Figure 4.11** The standard hexagonal domain $D$ with smallest radius containing a rectangular domain with length $a$, width $b$ and centred at $c$.

$$c = (c_1, c_2) \text{ and } r = \begin{cases} \frac{1}{2}(b + \frac{a}{\sqrt{3}}) & \text{if } b \geq \frac{a}{\sqrt{3}} \\ \frac{a}{\sqrt{3}} & \text{if } b < \frac{a}{\sqrt{3}} \end{cases}$$

Notice that the standard domain $D$ with smallest radius is not unique for the case $b < a/\sqrt{3}$, as the centre $c$ can be moved up or moved down.

## 4.4 An extended branch and bound framework

The Horst and Tuy branch and bound framework has been reviewed in Chapter 2. In [36] these authors showed that the algorithms of Pintér, and Zheng and Galperin are encompassed by this general framework. Somewhat surprisingly, the algorithm of Mladineo is also shown to sit beneath this umbrella, but the method used is not completely natural.

Our aim in this section is to slightly broaden the framework so that it more readily encompasses algorithms such as those discussed in this chapter. Motivation to alter the branch and bound framework springs from the observation that in the Piyavskii-Shubert algorithm. and multidimensional bisection, the natural cover at any stage is the projection onto the domain of the simplicial tops of the possibly overlapping simplex brackets. The idea is illustrated in Figure 4.12, for the case where $d = 1$.

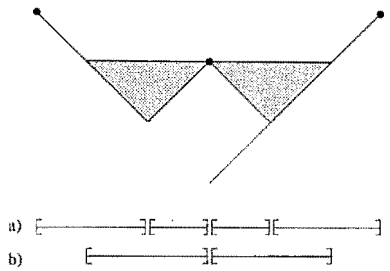



**Figure 4.12** The idea behind the new framework: after three evaluations (shown as heavy dots) the partition determined by the removal cones (used in [36]) is shown in (a), and the cover using simplex tops in (b).

Note that every finite partition is a finite cover, and moreover that every finite cover gives rise to a (not necessarily unique) finite partition. The language of covers, however, allows us to express multidimensional bisection as a branch and bound algorithm more conveniently than the language of partitions.

### A new branch and bound framework

We now present the branch and bound framework of Horst and Tuy in the language of covers, rather than partitions. We begin with a definition of a cover.

**Definition 4.4.1** *Let $D$ be a subset of $\boldsymbol{R}^d$ and $I$ be a finite index set. A set $\{C_i : i \in I\}$ of subsets of $D$ is said to be a cover of $D$ if $D \subseteq \cup_{i \in I} C_i$.*

Note that the cover sets may not intersect only on their boundaries. The set $C$ in $\mathbf{R}^d$ is termed *feasible* if $C \cap D \neq \phi$, and *uncertain* if it is not known whether it is feasible. We adopt the convention that the minimum taken over an empty subset of $\mathbf{R}$ equals $+\infty$.

**Initial step** ($k = 0$): The algorithm begins with a compact set $C_0$ covering $D$, or a subset of $D$ where $\min f(D)$ is realised. Set $\mathcal{C}_0 = \{C_0\}$, the initial cover. Associated with $C_0$ are bounds $\beta_0 = \beta(C_0)$ and $\alpha_0 = \alpha(C_0)$ such that

$$\beta_0 \leq \min f(D) \leq \alpha_0 = \min f(S_{C_0})$$

where $S_{C_0}$ is the (possibly empty) set of evaluation points made in $D$. If $\alpha_0 < \infty$, then choose $x^0$ such that $f(x^0) = \alpha_0$. If $\alpha_0 - \beta_0 \leq \epsilon$, then stop, else proceed to the iterative step.

**Iterative step** ($k = 1, 2, \ldots$): At the outset we have the finite cover of closed sets, $\mathcal{C}_{k-1}$, of the subset of $C_0$ still of interest. For every $C \in \mathcal{C}_{k-1}$ we have bounds $\beta(C)$ and $\alpha(C)$ satisfying

$$\begin{aligned} &\beta(C) \leq \min f(C \cap D) \leq \alpha(C) = \min f(S_C) && \text{if } C \text{ is known to be feasible, and} \\ &\beta(C) \leq \min f(C) && \text{if } C \text{ is uncertain} \end{aligned}$$

where $S_C$ is the possibly empty set of evaluation points in $C \cap D$, and the overall lower and upper bounds are defined as

$$\beta_{k-1} = \min\{\beta(C) : C \in \mathcal{C}_{k-1}\}, \quad \alpha_{k-1} = \min\{\alpha(C) : C \in \mathcal{C}_{k-1}\}$$

satisfying $\beta_{k-1} \leq \min f(D) \leq \alpha_{k-1}$. We now describe the four-stage "branch and banish, bound and banish" iterative step, based on the presentation in [35]. Figure 4.13 illustrates this procedure.

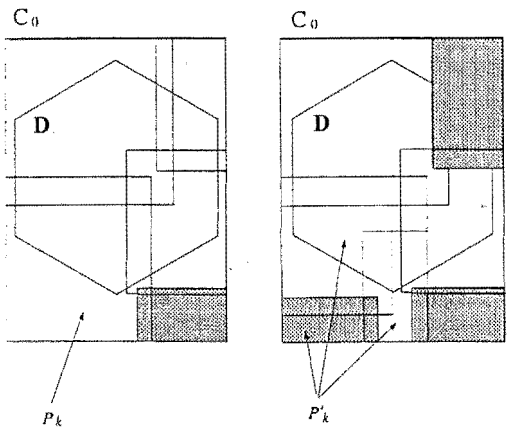



**Figure 4.13** An iteration step of the branch and bound procedure with rectangular covers of the hexagonal domain $D$. The shaded parts are covering sets banished in step $k$.

**Step 1. Branch:** Select a subset $\mathcal{P}_k$ of $\mathcal{C}_{k-1}$ and finitely recover each member of $\mathcal{P}_k$. Let $\mathcal{P}'_k$ be the set of all newly formed cover sets.

**Step 2. Banish:** Delete $C \in \mathcal{P}'_k$ if it lies outside $D$, or if it is known that $\min f(D)$ cannot occur over $C$. Let $\mathcal{C}'_k$ be the collection of cover sets remaining.

**Step 3. Bound:** Assign to each $C \in \mathcal{C}'_k$ which is known to be feasible bounds $\beta(C)$ and $\alpha(C)$ satisfying

$$\beta(C) \leq \min f(C \cap D) \leq \alpha(C) = \min f(S_C)$$

and to each uncertain $C \in \mathcal{C}'_k$ a bound $\beta(C)$ satisfying $\beta(C) \leq \min f(C)$. Here $S_C$ is the set of evaluation points in $C \cap D$. We assume that $\beta(C) \geq \beta(B)$ if $C \subseteq B \in \mathcal{C}_{k-1}$.

Let $\beta_k = \min\{\beta(C) : C \in \mathcal{C}'_k\}$ and $\alpha_k = \min\{\alpha(C) : C \in \mathcal{C}'_k\}$, the overall bounds, and if $\alpha_k < \infty$ let $x^k$ in $D$ be such that $f(x^k) = \alpha_k$.

**Step 4. Banish:** Delete all $C \in \mathcal{C}'_k$ not containing $x^k$ which are fathomed, that is, $\alpha_k \leq \beta(C)$. Let $\mathcal{C}_k$ be the collection of cover sets remaining.

If $\alpha_k - \beta_k \leq \epsilon$, then stop, else re-run the iterative step.

At this stage, $\beta_k \leq \min f(D) \leq \alpha_k$.

## Remarks

i) The cycling of the steps, placing "fathoming" last instead of first, does not alter the algorithm, and suits our purpose in the next section. It has, however, obliged us to add the phrase "not containing $x^k$" in Step 4 in order to exclude the possibility that all $C$'s are fathomed.

ii) We have assumed compactness throughout to ensure that the minima exist. This can be generalised to non-compact $C_0$ and infima, as in [35].

## 4.5 Multidimensional bisection within the branch and bound framework

We now show that the multidimensional bisection algorithm falls into this new branch and bound framework.

The natural domains $D$ for multidimensional bisection are the standard domains $D$ described in Definition 4.1.1(5). Such an algorithm begins by placing the minimum over such a $D$ in a simplex $C_0$, the top of the initial simplex bracket $T_0$ defined in Definition 4.1.2, projected onto the domain, $\mathbf{R}^d$. This simplex contains the region of $D$ still of interest. Figure 4.14 shows the situation for $d = 2$.

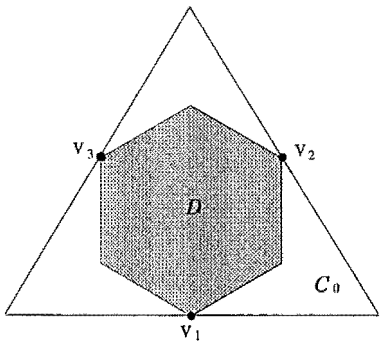



**Figure 4.14** The initial feasible set $D$, contained in the relaxed feasible set $C_0$. Shown here is the most conservative case, where all initial evaluations are equal. Variation in the function evaluations causes shrinkage of the initial cover set, $C_0$.

Evaluations of $f$ at the dual vertices $S_{C_0} = \{v_1, \ldots, v_{d+1}\}$ of $D$ provide an initial simplex $T(x_0, y_0, h_0)$ in $\mathbf{R}^{d+1}$ as constructed in Section 4.1. The height of the top of $T(x_0, y_0, h_0)$ forms the initial upper bound $\alpha_0 = \min\{f(v_i) : i = 1, \ldots, d+1\}$ while $y_0$, the base height, gives the initial lower bound $\beta_0 \leq \min f(D)$. Here $\alpha_0$ is always finite, so we immediately have an iteration point $x_0 \in D$ for which $f(x^0) = \alpha_0$. It is one of the vertices $v_i$ for which the evaluation is least.

At the start of the $k$th iteration a global minimum over $D$ is bracketed in a finite set of similar simplexes. The projection of these simplex tops onto the domain forms the cover $\mathcal{C}_{k-1}$. For the simplex in the system with projected top $C$, $\beta(C)$ equals the level of the simplex base, and $\alpha(C) = \min f(S_C)$ (where $S_C$ is the set of evaluation points in $C \cap D$) form lower and upper bounds for $\min f(C \cap D)$. A property of the algorithm is that for all feasible $C \in \mathcal{C}_{k-1}$ we have $\beta(C) \leq \min f(C \cap D) \leq \alpha(C)$. The first inequality follows since over $D$ the graph of $f$ remains on or above the sloping facets of the simplexes in the system.

We now describe how the deepest point multidimensional bisection iteration slots into the four stages of the branch and bound iterative step.

1. Branch: Select the $C \in \mathcal{C}_{k-1}$ corresponding to the simplex which is to be reduced, and any other cover sets influenced by elimination in this iteration. These sets constitute $\mathcal{P}_k$. The deepest point evaluation then yields a cover of $C$ which will be one of three types, according as the evaluation is above, on or below the simplex top. Figure 4.15 (a)-(c) shows these cases when $d = 2$.

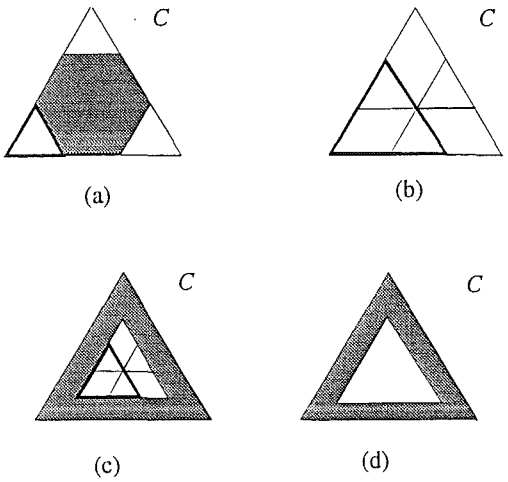



**Figure 4.15** The refinement of the cover set $C$ when the evaluation over the base is (a) above the top, (b) on the top, and (c) below the top. In cases (b) and (c) the refinement is a cover rather than a partition. Case (d) shows a cover set $C$, associated with the elimination phase, recovered using two sets.

Cover sets influenced by elimination will have their associated $d+1$ dimensional simplexes truncated, so give rise to the recovering shown in Figure 4.15 (d). This comprises two sets, shown shaded and unshaded.

2. Banish: The shaded regions in Figure 4.15, in (a) and (c), or their analogies for larger $d$, can now be deleted. This is possible since it is known that $\min f(D)$ cannot occur over these regions (see Section 4.1, System reduction). This leaves us with $\mathcal{C}_k'$. The geometry of multidimensional bisection would allow us to remove cover sets $C$ in $\mathcal{C}_0 \backslash D$. In practice we do not expend the effort, since Assumption 4.1.1 ensures that such sets are eventually fathomed in Step 4.

3. Bound: We assign to each new simplex $C$ in the cover the level of the base, $\beta(C)$. In multidimensional bisection this choice of $\beta(C)$ ensures that $\beta(C) \leq \min f(C \cap D)$ for each feasible set $C$, whether or not we know it to be feasible. That this inequality may not hold for infeasible $C$ is not a concern, since infeasible cover sets for which it does not hold will be more quickly fathomed in Step 4, a desirable outcome. With the annular shaded region in (d) we associate a $\beta$ value of $\alpha_k$, the level of the lowest evaluation to date. We know that $\min f(D)$ cannot occur on such a $C$, so this ensures that it is fathomed in Step 4. To each cover set we can assign $\alpha(C) = \min f(S_C)$, where $S_C$ is the set of evaluation points in $C \cap D$. Certainly it follows that $\min f(C \cap D) \leq \alpha(C)$. The evaluation point $x^k$ in $D$ is chosen such that $f(x^k) = \alpha_k$.

   These three steps together correspond to the reduction step, $\mathcal{R}$. They are viewed simultaneously in multidimensional bisection, but can be linearly ordered in the way just described.

4. Banish: This fourth step corresponds to the elimination step, $\mathcal{E}$. All shaded regions of the type shown in Figure 4.15 (d) are removed.

This completes the demonstration that the multidimensional bisection algorithm with the deepest point strategy is an example of the new framework. A fine point should be acknowledged here: any (necessarily singleton) $C \neq \{x^k\}$ for which $\beta(C) = \alpha_k$ would be eliminated in Step 4. Thus we are capturing a very slight modification of multidimensional bisection. The algorithm just described, however, still is such that $f(x^k) \downarrow \alpha$.

Incorporation of spherical reduction would alter only the refinement of $\mathcal{P}_k$. Any complete reduction, or a reduction strategy which involved more than a single evaluation (such as that in the next section) would necessitate overlaying the covers generated by reduction and elimination. These algorithms still follow the pattern of the branch-and-bound format.

Similarly we can demonstrate that FMBA, with or without the acceleration strategies, also falls into the new branch and bound framework. Therefore, the multidimensional bisection algorithm is an example of the new framework.

### Range convergence

We turn now to the convergence of the multidimensional bisection algorithms. We review the convergence of the branch and bound algorithm in terms of covering presented in Section 4.4, and show how this relates to the convergence of DMBA.

For the $\alpha_k$ and $\beta_k$ of Section 4.4, evidently $\alpha_k$ is non-increasing and $\beta_k$ non-decreasing. Thus $\lim \alpha_k = \alpha$ and $\lim \beta_k = \beta$ necessarily exist, and $\beta \leq \min f(D) \leq \alpha$. Following Horst and Tuy we say that an infinite procedure (one for which $\alpha_k \neq \beta_k$ for all $k$) converges if $\alpha_k - \beta_k \to 0$, as $k \to \infty$, whence

$$\alpha = \lim_k f(x^k) = \beta = \min f(D).$$

We now restate in appropriate form the convergence conditions for an in-

finite branch and bound procedure in terms of covering. These will ensure that $\alpha_k - \beta_k \to 0$.

Definitions

1. A bounding procedure is termed *consistent* if at the start of each iteration every non-degenerate cover set can be refined, and any decreasing sequence $C_{k_q}$ coming from successively refined covers satisfies

$$\lim_q \left(\alpha_{k_q} - \beta(C_{k_q})\right) = 0 \ .$$

2. A selection procedure is termed *complete* if for every $C \in \cup_{p=1}^{\infty} \cap_{k=p}^{\infty} \mathcal{C}_k$ we have $\min f(C \cap D) \geq \alpha$ .

3. A selection procedure is termed *bound improving* if, for each $k$, there exists an $l \geq k$ for which

$$\text{argmin}\{\beta(C) : C \in \mathcal{C}_k\} \cap \mathcal{P}_l \neq \phi \ .$$

That is, at each iteration one covering element where $\beta_k$ occurred is later selected for refinement.

The following theorem, which we restate from Theorem 2.2.1 and 2.2.2 in the language of coverings, then follows, with the proofs as in [35].

**Theorem 4.5.1** *In an infinite broadened branch and bound procedure involving covering, suppose that the bounding operation is consistent. It follows that*

*i) if the selection is complete, then*

*a)* $\alpha = \min f(D)$,

*b) if* $f$ *is continuous, then every accumulation point* $x$ *of* $\{x^k\}$ *is such that* $f(x) = \min f(D)$.

*ii) if the selection is bound improving, then the procedure is convergent, so*

$$\alpha = \min f(D) = \beta.$$

We now use this theorem to discuss the convergence of DMBA. The deepest point strategy of multidimensional bisection ensures that all deepest simplexes in the system at the end of each multidimensional bisection iteration are eventually reduced. This is readily shown to be equivalent to the bound improving condition. For multidimensional bisection, a selection procedure which is bound improving also is such that the bounding procedure is consistent. This is shown in Theorem 4.1.1 (ii). When the bounding in the algorithm is consistent, the selection is complete [35, p.127]. In DMBA, the location of the least evaluation to date is chosen as the iteration point, $x^k$. Assumption 4.1.1 ensures that eventually all such points are feasible, so that the accumulation points of this sequence coincide with those of the sequence of least feasible evaluation points. It follows from Theorem 4.1.1 (ii) that multidimensional bisection, with the deepest point strategy, is "range" convergent, or $f(x^k) \rightarrow f_*$.

## 4.6 On the localisation produced by multidimensional bisection

Basso in [8] discussed the problem of determining the localisation of all global optima of a univariate function as well as the optimal value of such a function, that is, problem $Q$ for $d = 1$. He also studied the behaviour of the Piyavskii-Shubert algorithm. It was pointed out in [8] that the deepest point reduction strategy of the Piyavskii-Shubert algorithm does not ensure "domain" convergence of the algorithm. We clarify this statement using the following notation described in Chapter 1. Denote by

i) $A$, the accumulation points of the iteration points,

ii) $E$, the points in $D$ where $\min f(D)$ is realised,

iii) $L_\infty$, the set $\cap_{k=0}^{\infty} L_k$, where $L_k = \cup\{C : C \in \mathcal{C}_k\}$ is the "localisation" at the $k$th iteration.

The Piyavskii-Shubert algorithm ensures that $A \subseteq E \subseteq L_\infty$ for the univariate case, as this algorithm is a special case ($d = 1$) of DMBA, whence Theorem 4.1.1 iii) follows. The function pictured in Figure 4.16, adapted from Basso, illustrates that for the deepest point algorithm we cannot guarantee equality in the inclusions, $A \subseteq E \subseteq L_\infty$.

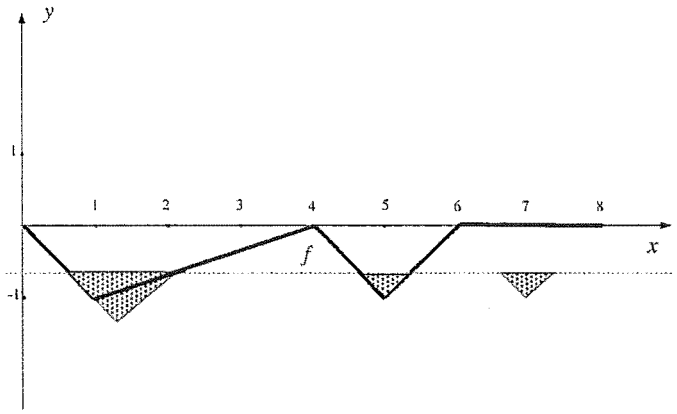



**Figure 4.16** Modified Basso function: For the $f$ shown, the solution set $E = \{1, 5\}$. The Piyavskii-Shubert algorithm has accumulation point set $A = \{1\}$ and final localisation $L_\infty = \{1, 5, 7\}$, showing that in general equality is not the case for the inclusions $A \subseteq E \subseteq L_\infty$. A typical system is shaded in the figure.

A typical system is sketched over the function after four iterations of the Piyavskii-Shubert algorithm. Iteration points stay in the neighbourhood of $x = 1$, with the points $(5, -1)$ and $(7, -1)$ remaining in all systems. Note that $5 \in E$ but is not an accumulation point of the iteration points, and $7 \in L_\infty$ but is not in $E$. So, $A = \{1\}, E = \{1, 5\}$ and $L_\infty = \{1, 5, 7\}$. Thus both inclusions are proper.

To obtain domain convergence, that is, $A = E = L_\infty$, we need to remove such an unwanted point 7 from $L_\infty$ and make point 5, a minimiser, an accumulation point.

The least Lipschitz constant of the function $f$ is used in the above example, the value $M = 1$. One way to make every solution point in $E$ be an accumulation point is to use a slightly enlarged Lipschitz constant in the problem. Then the relation $A = E$ is ensured by the convergence result of the Pintér algorithm described in Chapter 2, as the Piyavskii-Shubert algorithm is a special case of the Pintér algorithm under such a Lipschitz constant modification.

Basso discussed in [8] a necessary and sufficient condition on the lower bounding function to ensure that $A = E = L_\infty$ for the case $d = 1$. The following two theorems are drawn from [8]. We extend the second into Theorem 4.6.3.

**Theorem 4.6.1** (*Basso, [8, Theorem 1]*) *Let $f$ be a continuous real function on $[a, b]$ with $\min_{x\in[a,b]} f(x) = f_*$. Let $\{F_k\}$, for $k \geq 0$, be a sequence of lower bounding functions of $f$, $\{\alpha_k\}$ be a sequence of real numbers and $L_k = \{x \in [a, b] : F_k \leq \alpha_k\}$ be such that:*

*i)* $\alpha_k \geq f_*$,

*ii)* $\lim_{k\to\infty} \alpha_k = f_*$,

*iii)* $\lim_{k\to\infty} \inf_{x\in L_k} F_k(x) = f_*$.

*Then, a necessary and sufficient condition for*

$$L_\infty = \cap_{k=0}^{\infty} L_k = E = \{x \in [a, b] : f(x) = f_*\}$$

*is*

$$\lim_{k\to\infty} F_k(x) = f(x) \quad \textit{for any} \quad x \in L_\infty.$$

That the above condition ensures $L_\infty = A$ is also discussed by Basso. He proposed two ways to modify of the Piyavskii-Shubert algorithm, named the *block-sequential* or *Jacobi type* method and the *Gauss-Seidel* method.

The block-sequential method at step $k$ evaluates the function simultaneously at the deepest point in each disjoint localisation of $L_k$, instead of evaluating the function at only *one* deepest point as in the Piyavskii-Shubert algorithm. This procedure consists precisely in simultaneously modifying the lower bounding function in each and every disjoint interval. The Gauss-Seidel type method updates the best function value $f_*^k$ at *each* evaluation during the $k$th iteration instead of updating $f_*^k$ after a full iteration as adopted in the block-sequential method. Alternation of the Jacobi and Gauss-Seidel methods forms the *chaotic* method.

All three methods satisfy the asymptotic convergence condition of Theorem 4.6.1 and therefore guarantee the localisation convergence of the original problem, problem $Q$. The following theorem states that localisation convergence can be extended to include the set of accumulation points $A$ by using the block-sequential method for solving problem $Q$ for the case $d = 1$.

**Theorem 4.6.2** (*Basso, [8, Theorem 2 and Corollary 2]*) *Let $f \in L(M)$ on $[a, b]$ and $\{F_k\}$ for $k \geq 0$ be a sequence of lower bounding functions of $f$. Let $\{\alpha_k\} = \{f_*^k\}$ be a sequence of real numbers and $L_k = \{x \in [a, b] : F_k \leq \alpha_k\}$, the sequence of localisations generated by the block-sequential method. Then*

$$A = E = L_\infty.$$

As the deepest point algorithm coincides with the Piyavskii-Shubert algorithm when $d = 1$, DMBA with or without accelaration strategies also cannot guarantee localisation convergence without modification.

Recall that for a consistent and bound improving multidimensional bisection (DMBA satisfies these conditions) we have $A \subseteq E \subseteq L_\infty$. This follows from Theorem

4.1.1.

Before giving the main theorem of this section, we define a block-sequential reduction strategy for multidimensional bisection which extends Basso's terminology of block-sequential to the case when $d \geq 1$.

**Definition 4.6.1** *A system reduction strategy $\mathcal{R}(\mathcal{S})$ defined in Definition 4.1.4 is called block-sequential if the evaluations of the function $f$ are chosen simultaneously in each disjoint component of the localisation $L_k$ of the system $\mathcal{S}$.*

Notice that in block-sequential reduction, the size of the evaluation index $I$ may vary from step to step. It is greater than one but may be less than $|J|$. Therefore a multidimensional bisection algorithm with block-sequential reduction is sandwiched between DMBA and FMBA.

We now extend Theorem 4.6.2 to higher dimensions. That is, we show that the multidimensional bisection algorithm equipped with the block-sequential reduction strategy guarantees that $A = E = L_\infty$. This result will hold for all four forms $A, A^c, A^s$, and $A^{cs}$ described in this chapter. Observe that this is the convergence property FMBA enjoys, but block-sequential reduction requires a weaker condition on the number of evaluation points in an iteration, as described above.

**Theorem 4.6.3** *For the multidimensional bisection algorithm, with block-sequential reduction and any combination of complete and spherical reduction, then*

$$A = E = L_\infty.$$

Proof: Block-sequential reduction is certainly bound-improving, so it follows as before that $A \subseteq E \subseteq L_\infty$. Hence it suffices to take $x \in L_\infty$ and show that $x \in A$.

From Theorem 4.1.1 (i) we know that $\alpha_k \downarrow f_*$ and $\beta_k \uparrow f_*$, where $f_* = \min f(D)$. Thus the only point over $x$ eventually remaining in the bracket is $(x, f_*)$. We consider two cases.

Case 1: $(x, f_*)$ is eventually only in simplexes with apex at the point $(x, f_*)$

Since $\alpha_k \downarrow f_*$ as $k \to \infty$, eventually $(x, f_*)$ lies in an isolated system simplex, of variation as small as we please. If $f(x) > f_*$, an evaluation at $x$, ensured by the assumption of block-sequential reduction, would remove $(x, f_*)$ from the system. Since $x \in L_\infty$ this provides a contradiction, so $f(x) = f_*$, and $x \in E$. It is also evident that eventually the system must include the degenerate simplex $\{(x, f_*)\}$. Block sequential reduction ensures that this isolated simplex will be evaluated in every later full iteration. Hence $x \in A$.

Case 2: $(x, f_*)$ is always in some simplex with apex at height less than $f_*$.

Since $\beta_k$ increases to $f_*$, $(x, f_*)$ must lie in a strictly nested sequence of simplexes, $T_k = T(x_k, y_k, h_k)$, as the algorithm progresses. Note that $h_k$ must decrease to zero.

For incomplete reduction, with or without spherical reduction, any raising of the apex of $T_k$ must occur through an evaluation at $x_k$. Thus $x \in A$.

For complete reduction, with or without spherical reduction, the apex of $T_k$ can be raised through an evaluation at a point other than $x_k$. We now show, however, that given a neighbourhood $U$ of $x$, and for $T_k$'s with projected top inside $U$, there can be at most finitely many evaluations outside $U$ which raise the level of these simplexes. Thus $x \in A$.

Suppose that $\{z_l\}$ is a sequence of evaluation points outside $U$, each of which raises the level of one of these simplexes. Then there exists an $\epsilon > 0$ such that $f(z_l)$ exceeds $f_* + \epsilon$ for each $l$. Since $C_0 - U$ is compact, the sequence of evaluation points $\{z_l\}$ has an accumulation point, $z$. Since $f$ is continuous, $f(z) > f_*$. But $z \in A \subseteq E$, so $f(z) = f_*$, a contradiction. This completes the proof.

□

Remarks

i) In trials of the algorithms using deepest point reduction, we have noted the occurrence of points of the type occurring at $x = 7$ in the example of Figure 4.16. With block-sequential reduction such a point would disappear in an early iteration. The result of Theorem 4.6.3 suggests that every so often a block-sequential iteration should be run to remove such stray points.

ii) The proof of Theorem 4.6.3 reveals that the behaviour exhibited in Figure 4.16 at $x = 1$ and $x = 5$ illustrates the only two ways in which a point can remain forever in the system.

CHAPTER 5

# A stochastic method: pure localisation search

## 5.1 Introduction

The deterministic Lipschitz based algorithms discussed in the previous chapters, for solving global optimisation problems $P$, $P'$, $Q$ and $Q'$ possess guaranteed convergence properties. In this chapter, we investigate stochastic algorithms for solving problem $P'$. Is there a stochastic approach which can solve problem $P'$, and which possesses polynomial complexity in terms of the number of iterations required as the dimensions increases? We explore this question in this chapter.

Two stochastic algorithms, pure random search (PRS) and pure adaptive search (PAS), for solving the global optimisation problem $P'$ have been reviewed in Chapter 2. Pure random search is easy to implement, as the next evaluation point is always chosen on the initial feasible region according to a uniform distribution. But it is not an effective approach as it does not use information from previous evaluation points and values. The expected number of iterations in order to be within a given tolerance of the global minimum also increases exponentially in the dimension of the problem, as described in [81]. Pure adaptive search has been introduced and discussed in [49] and [81]. In PAS, the next point is chosen according to a uniform distribution on the improving region, or "inside the level set", of the feasible space. In [81] it was shown that when pure adaptive search is applied

to global mathematical programs satisfying the Lipschitz condition, the expected number of iterations in order to be within a given tolerance of the global minimum increases at most linearly in the dimension of the problem, a desirable complexity result. A difficulty which immediately arises is that pure adaptive search appears to be hard to realise efficiently in practice. Encouragement, however, comes from the observation that several other practical random search algorithms have reported linearity in dimension, for example [72], although only for convex programs.

Pure adaptive search can be implemented, albeit very inefficiently, by running pure random search and accepting only those points which provide improved function evaluations. Two attempts have already been made to provide a more efficient implementation. These are the Improving Hit-and-Run algorithm [82], and the Hide-and-Seek algorithm [64]. For a Lipschitz continuous objective function, algorithms for solving problem $P'$ can utilise the facts described in Section 2.1: at each iteration, regions which cannot contain the global minima can be stripped away from the domain. The remaining region of interest, the "localisation" $L_k$, to use the term we used in the previous chapters, is a set which properly contains the level set of PAS at each iteration. From this viewpoint, we may raise this question: is there an efficiently implementable algorithm, based on a stochastic variant of the Piyavskii-Shubert algorithm, which can realise the desirable complexity of PAS? The procedure proposed which might meet these requirement is: choose the evaluation point on an enlargement of the level set of pure adaptive search. The enlargement must be accessible in practice, yet small enough to retain the desirable properties of PAS. Such an enlargement is provided by some of the Lipschitz based deterministic algorithms for mathematical programs, such as the Piyavskii-Shubert algorithm, the Mladineo algorithm and the Wood algorithm reviewed in Chapter 2.

We develop this chapter as follows. In Section 5.2, somewhat adaptive search

(SAS) is introduced. SAS is a relaxation of pure adaptive search, and is more likely to be efficiently implementable, yet still possesses the desirable complexity of PAS. Theorem 5.2.1 states the linear complexity result, and discusses the special case of "$\rho$-adaptive" search. In Section 5.3 a modification of the Piyavskii-Shubert algorithm (PS) to a stochastic search, pure localisation search (PLS), is proposed and the spherical and simplicial realisations of this algorithm are discussed. A comparison of the performance of PRS, PAS, PLS and PS for a set of univariate test functions is provided in Section 5.4. The link between SAS and PLS is discussed in Section 5.5, and numerical results confirming the theoretical results are shown. Much of the material in this chapter has appeared in [7].

## 5.2 Somewhat adaptive search

Consider the global optimisation problem $P$ described in Chapter 1. That is, find the global minimum of an objective function $f$ on a convex, compact full-dimensional domain $D$ of $\mathbf{R}^d$ for $f \in L(M)$. Denote the optimal solution by $(x_*, y_*)$, where $x_* \in \arg\min_{x \in D} f(x)$ and $y_* = f(x_*)$. It is convenient to define $y^* = \max_{x \in D} f(x)$. We do not require that a unique minimum point should exist. If there is more than one, we choose $x_*$ arbitrarily.

Consider stochastic sequential search procedures whose aim is to locate $(x_*, y_*)$. The sample path of evaluation points is denoted by $X_1, X_2, \ldots$ and the associated function values $Y_1, Y_2, \ldots$. Epoch $i > 1$ is said to be a *record* of the sequence $\{Y_k\}$ if $Y_i < \min\{Y_1, \ldots, Y_{i-1}\}$ or $Y_i = y_*$. For technical reasons it is convenient to include the latter condition. Epoch $i = 1$ is always considered to be a record. The corresponding value $Y_i$ is called a *record value*. For $k \geq 1$ let $R(k)$ denote the epoch of the $k$th record value of the sequence of evaluations. For $k \geq 2$, the number of iterations from the $(k-1)$st to the $k$th record is denoted by $I_k$. Thus $I_k = R(k) - R(k-1)$.

The expected value of a random variable $V$ is denoted by $E(V)$. Two sequences of random variables are said to be stochastically equivalent if they have the same joint probability distribution function.

We pause to recall the definition of the PAS algorithm for solving problem $P$:

**Definition 5.2.1** *Pure Adaptive Search (PAS)*

***Initial Step:*** *Set $k = 0$ and $S_0 = D$.*

***Iterative Step:*** *Increment $k$*

*i) Select evaluation point.*

*Choose $x_k$ uniformly distributed on $S_{k-1}$ if non-empty,*

*else set $x_k = x_{k-1}$*

*Set $w_k = f(x_k)$.*

*ii) Update localisation.*

*Set $S_k = f^{-1}(-\infty, w_k)$*

***Stopping Criterion:*** *Stop if a stopping criterion is met, else return to the iterative step.*

Here $f^{-1}(-\infty, w_k) = \{x \in D : -\infty < f(x) < w_k\}$. A stopping criterion can be chosen as, for instance, that a pregiven iteration number is met.

The definition of somewhat adaptive search is an attempt to keep the PAS virtue of linear complexity in dimension while at the same time allowing room to construct practical algorithms. The algorithms require that two conditions should hold. The first allows the algorithm to mark time between records, but not for too long, while the second insists that the quality of the records be as good as those of PAS. The first condition gives the space needed to implement the algorithm, while together they ensure that the "linearity in dimension" drawcard of PAS is retained.

**Definition 5.2.2** *A stochastic sequential search algorithm for solving problem $P$ is termed somewhat adaptive search (SAS) if the following two conditions are satisfied:*

*i) There exists a bound $\beta \geq 1$ such that $E(I_k) \leq \beta$ for all $k$, and*

*ii) $\{Y_{R(k)} : k = 1, 2, \ldots\}$ is stochastically equivalent to $\{W_k : k = 1, 2, \ldots\}$, the sequence of records of PAS.*

In [81], for a non-constant function, Zabinsky and Smith define the *relative improvement* associated with an evaluation $y > y_*$ as $z = (y^* - y)/(y - y_*)$. In order to state a similar result for SAS, we extend the language of [81] to:

$N_{SAS}(z)$ = the number of iterations of SAS achieving a relative improvement of $z$ or less

$N^*_{SAS}(y)$ = the number of iterations of SAS required to achieve a value of $y$ or lower.

The wording of the above must be heeded carefully. The first expression is the number of iterations *achieving* something, while the second is the number *required* before something is achieved.

The corresponding expressions for PAS are denoted by $N_{PAS}(z)$ and $N^*_{PAS}(y)$. The following relation is easy to show and worth noting:

$$E\left[N^*_{SAS}(y)\right] = 1 + E\left[N_{SAS}(z)\right]$$

where $z = (y^* - y)/(y - y_*)$. The following theorem gives a comparison of the expectation of $N^*_{SAS}(y)$ and $N^*_{PAS}(y)$ and a performance bound on the expected numbers of the iterations for solving problem $P'$.

**Theorem 5.2.1** *Consider all global optimisation problems $P'$ over a convex feasible region $D$ in $\boldsymbol{R}^d$ with diameter at most $\delta$, and all functions $f \in L(M)$. Suppose an*

*algorithm is SAS for this class of problems, and the bound $\beta$ in Definition 5.2.2 $(i)$ is at most $B$. Then*

$$E\left[N^*_{SAS}(y)\right] \;\leq\; \beta\; E\left[N^*_{PAS}(y)\right] \;\leq\; B + \left[B \ln(M\delta/(y - y_*))\right] d$$

*That is, the bound is a linear function of the dimension $d$ of the problem.*

Proof: For a constant function some of the above terms are undefined. In this case it is easy to check that SAS is actually PAS and the theorem follows trivially. For a non-constant function, since the $Y_{R(k)}$ are stochastically equivalent to the $W_k$ it follows that $N_{SAS}(z) = I_1 + I_2 + \ldots + I_{N_{PAS}(z)}$. Then

$$\begin{aligned} E\left[N_{SAS}(z)\right] &= E\left\{E\left[I_1 + \ldots + I_{N_{PAS}(z)} \middle|\; N_{PAS}(z) \;\text{ fixed}\right]\right\} \\ &\leq E\left[\beta N_{PAS}(z)\right] \\ &= \beta E\left[N_{PAS}(z)\right] \end{aligned}$$

Converting this into a result about $N^*_{SAS}$, we see

$$\begin{aligned} E\left[N^*_{SAS}(y)\right] &\leq E\left[N^*_{SAS}(y)\right] - 1 + \beta \\ &= E\left[N_{SAS}\left(\frac{y^* - y}{y - y_*}\right)\right] + \beta \\ &\leq \beta\; E\left[N_{PAS}\left(\frac{y^* - y}{y - y_*}\right)\right] + \beta \\ &= \beta\; E\left[N^*_{PAS}(y)\right], \quad \text{as required.} \end{aligned}$$

From Theorem 5.3 of [81], we have that

$$E\left[N^*_{PAS}(y)\right] \;\leq\; \left[\ln(M\delta/(y - y_*))\right] d, \quad \text{therefore}$$

$$\begin{aligned} \beta\; E\left[N^*_{PAS}(y)\right] &\leq \beta + \left[\beta \ln(M\delta/(y - y_*))\right] d \\ &\leq B + \left[B \ln(M\delta/(y - y_*))\right] d \quad \text{as required.} \end{aligned}$$ □

The above theorem asserts that SAS, a relaxation of PAS, possesses the "linearity in dimension" complexity property. The complexity is measured here in

terms of the expected number of iterations to be within $y - y_*$ of the global minimum $y_*$.

The following definition of "$\rho$-adaptive search", a special case of SAS, can be described informally as follows: at each iteration the conditional probability that it behaves as PAS is at least $\rho$.

**Definition 5.2.3** *Let $0 \leq \rho \leq 1$. A stochastic sequential search for solving problem $P$ is termed* $\rho$-adaptive *if for each iteration $k$, and for all sample paths $x_1, \ldots, x_{k-1}$,*

$$Prob[X_k \text{ is distributed uniformly in the improving region} \mid x_1, \ldots, x_{k-1}] \geq \rho$$

In this framework, PRS is 0-adaptive and PAS 1-adaptive. This language gives us a way of describing a spectrum of algorithms between these two extremes. It is readily seen that a non-degenerate $\rho$-adaptive algorithm ($\rho \neq 0$) is always SAS by checking the conditions of Definition 5.2.2. For condition i), the definition of $\rho$-adaptivity ensures that after any iteration $k$, and independent of the sample path, the probability of a record is greater than or equal to $\rho$. Thus $E(I_k)$ is less than or equal to the mean of a geometric distribution, with parameter $\rho$. Thus $E(I_k) \leq 1/\rho$, for all $k$. Condition ii), that the $Y_{R(k)}$ are stochastically equivalent to the $W_k$, follows via a straightforward modification of [81, Lemma 3.1].

Some algorithms which are related to $\rho$-adaptive search include those of Bogomolov and Karmanov [9], Denisov [14], Devroye [15], Marti [40], Solis and Wets [72], Pintér [51, 53] and Zhigljavsky [87].

## 5.3 Pure localisation search

We now introduce a readily implemented algorithm for solving problem $P'$. In spirit, the algorithm is a probabilistic analogue of the Piyavskii-Shubert algorithm.

Initially we present the algorithm in a general setting.

The central idea is the following. An exact tracking of the level set of PAS is an impossible task. Tracking a superset of it is not. Certain "removal" algorithms in the literature, for example [5, 43, 61, 69, 78], while deterministic, do yield a localisation $L_k$ for the level set at each iteration.

**Definition 5.3.1** *Pure Localisation Search, PLS.*

***Initial Step:*** *Set* $k = 0$ *and* $L_0 = D$.

*Set* $\alpha_0 = \infty$.

***Iterative Step:*** *Increment* $k$

*i) Select evaluation point.*

*Choose* $x_k$ *uniformly on* $L_{k-1}$ *if non-empty,*

*else set* $x_k = x_{k-1}$.

*Set* $y_k = f(x_k)$.

*ii) Update localisation.*

$$\textit{Set } \alpha_k = \begin{cases} y_k, & \textit{if } y_k < \alpha_{k-1} \\ \alpha_{k-1}, & \textit{otherwise.} \end{cases}$$

*Set* $L_k = L_{k-1} - R_k$, *where the removal region* $R_k$ *is such that* $R_k \subseteq D - f^{-1}(-\infty, \alpha_k)$.

***Stopping Criterion:*** *Stop if a stopping criterion is met, else return to the iterative step.*

Figure 5.1 illustrates the idea of PLS for $d = 2$.

We define $D_y = f^{-1}(-\infty, y)$. Observe that the special case of PLS with $R_k = \phi$ is PRS, while PLS becomes PAS when $R_k = D - D_{\alpha_k}$, so $L_k = D_{\alpha_k}$. It follows from Theorem 5.3.2 of this section that PLS converges with probability one.

An important observation concerning any PLS is that $L_k \supseteq D_{\alpha_k}$, or in other words, the localisation contains the improving set.

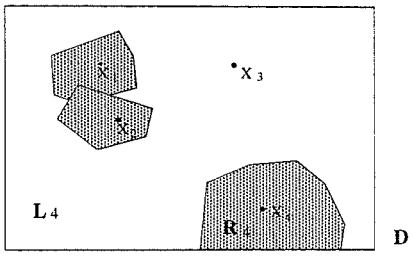



**Figure 5.1** Pure localisation search after four evaluations at $x_1, \ldots, x_4$, with $f(x_3) = \min_{1 \le i \le 4} f(x_i)$. The shaded part is the removed region and the unshaded part the localisation $L_4$.

That PLS always has the second property of SAS is shown in the next theorem.

**Theorem 5.3.1** *For PLS applied to global optimisation problem P, the stochastic process of PLS record values is equal in distribution to the stochastic process of PAS record values. That is*

$$\{Y_{R(k)} : k = 1, 2, \ldots\} \sim \{W_k : k = 1, 2, \ldots\}$$

Proof: The proof is an extension of [81, Lemma 3.1]. First we show that the conditional distributions are equal. Let $k$ be any iteration and take $y_* \le y < y' \le y^*$. Note that

$$P[W_{k+1} < y | W_k = y'] =$$

$$P[Y_{R(k)+j} < y | Y_{R(k)} = y' \text{ and } Y_{R(k)+1} \ge y' \ldots Y_{R(k)+j-1} \ge y' \text{ and } Y_{R(k)+j} < y']$$

since both sides equal $\lambda(D_y)/\lambda(D_{y'})$. Here $\lambda$ denotes Lebesgue measure on $\mathbf{R}^d$. Then we have

$$P[Y_{R(k+1)} < y | Y_{R(k)} = y']$$

$$= P[Y_{R(k)+1} < y | Y_{R(k)} = y'] + P[Y_{R(k)+1} \ge y' \text{ and } Y_{R(k)+2} < y | Y_{R(k)} = y'] + \ldots$$

$$
\begin{aligned}
&= P[Y_{R(k)+1} < y' | Y_{R(k)} = y'] \cdot P[Y_{R(k)+1} < y | Y_{R(k)} = y' \text{ and } Y_{R(k)+1} < y'] \\
&\quad + P[Y_{R(k)+1} \geq y' | Y_{R(k)} = y'] \cdot P[Y_{R(k)+2} < y' | Y_{R(k)} = y' \text{ and } Y_{R(k)+1} \geq y'] \cdot \\
&\quad P[Y_{R(k)+2} < y | Y_{R(k)} = y' \text{ and } Y_{R(k)+1} \geq y' \text{ and } Y_{R(k)+2} < y'] + \ldots \\
&= [p_1 + (1-p_1)p_2 + (1-p_1)(1-p_2)p_3 + \ldots] \, P[W_{k+1} < y | W_k = y']
\end{aligned}
$$

where $p_i = P[Y_{R(k)+i} < y' \mid Y_{R(k)} = y' \text{ and } Y_{R(k)}, \ldots, Y_{R(k)+i-1} \geq y']$, the probability that the first record after the $k$th record occurs at epoch $R(k)+i$. Now the sequence $\{p_i\}$ is bounded away from zero, since $p_i \geq \lambda(D_{y'})/\lambda(D) > 0$ for all $i$. It follows by an elementary argument that $p_1 + (1-p_1)p_2 + (1-p_1)(1-p_2)p_3 + \ldots = 1$. In fact, let $q_i = 1 - p_i$, then $0 \leq q_i < 1$, so

$$
\begin{aligned}
& p_1 + (1-p_1)p_2 + (1-p_1)(1-p_2)p_3 + \ldots + (1-p_1)(1-p_2)\ldots(1-p_{n-1})p_n \\
= \;& 1 - q_1 + q_1(1-q_2) + q_1q_2(1-q_3) + \ldots + q_1q_2\ldots q_{n-1}(1-q_n) \\
= \;& 1 - q_1q_2\ldots q_n \\
\longrightarrow \;& 1 \quad \text{as} \quad n \to \infty. \quad \text{Thus}
\end{aligned}
$$

$$
P[Y_{R(k+1)} < y | Y_{R(k)} = y'] \;=\; P[W_{k+1} < y | W_k = y'].
$$

We now use induction to show that the unconditional distributions are equal. By convention, $R(1) = 1$ and from the definition of PLS, $P[Y_1 < y] = P[W_1 < y]$ for all $y$, for $y_* \leq y \leq y^*$. Hence $Y_{R(1)} \sim W_1$.

Now consider $k > 1$ and suppose that $Y_{R(i)} \sim W_i$ for $i = 1, 2, \ldots, k$. Then, for all $y_* \leq y \leq y^*$, we have

$$
\begin{aligned}
P[Y_{R(k+1)} < y] &= \int_{y_*}^{y^*} P[Y_{R(k+1)} < y | Y_{R(k)} = t] dF_{Y_{R(k)}}(t) \\
&= \int_{y_*}^{y^*} P[W_{k+1} < y | W_k = t] dF_{W_k}(t) \\
&= P[W_{k+1} < y]
\end{aligned}
$$

The second equality follows using the equality of conditional distributions and the induction hypothesis.

By induction it follows that the two sequences are equal in marginal distribution, hence in joint distribution, as required. □

The way in which PLS is sandwiched between PRS and PAS is made clear in the next theorem and its immediate corollary.

**Theorem 5.3.2** *Fix $f$ as in problem P, and fix a relative improvement level, $z > 0$. Let $N_{PAS}(z)$, $N_{PLS}(z)$ and $N_{PRS}(z)$ be the number of iterations of PAS, PLS and PRS respectively achieving a relative improvement of $z$ or less. Then*

$$P[N_{PAS}(z) < k] \geq P[N_{PLS}(z) < k] \geq P[N_{PRS}(z) < k]$$

**Corollary 5.3.1**

*i)* $E[N_{PAS}(z)] \leq E[N_{PLS}(z)] \leq E[N_{PRS}(z)]$

*ii)* $N^p_{PAS}(z) \leq N^p_{PLS}(z) \leq N^p_{PRS}(z)$ *where $N^p_{alg}(z)$ is the number of iterations of the algorithm required to achieve a relative improvement of $z$ with probability not less than $p$.*

Proof of Theorem: Let $y$ correspond to a relative improvement of $z$. Then

$$\begin{aligned} P[N_{PAS}(z) < k] &= P[W_k \leq y] \\ &= P[Y_{R(k)} \leq y], \quad \text{by Theorem 5.3.1} \\ &\geq P[N_{PLS}(z) < k] \end{aligned}$$

since if PLS achieves a relative improvement of $z$ before the $k$th iteration, $Y_k \leq y$, whence $Y_{R(k)} \leq y$ .

In order to show that $P[N_{PLS}(z) < k] \geq P[N_{PRS}(z) < k]$ we now show that $P[N_{PLS}(z) > k-1] \leq P[N_{PRS}(z) > k-1]$. Note that $i = N_{PLS}(z) + 1$ is the first

epoch such that $x_i \in D_y$. Thus, for any $k$, $k = 1, 2, \ldots$,

$$\begin{aligned} & P[N_{PLS}(z) > k-1] \\ = \; & P[x_1 \notin D_y \text{ and } x_2 \notin D_y \ldots \text{ and } x_k \notin D_y] \\ = \; & P[x_1 \notin D_y]\; P[x_2 \notin D_y | x_1 \notin D_y]\; \ldots\; P[x_k \notin D_y | x_1, \ldots, x_{k-1} \notin D_y] \end{aligned}$$

Consider a PLS sample path with first $j-1$ domain points not in $D_y$, where $j \in \{1, \ldots, k\}$. Then $\alpha_{j-1} \geq y$, so $D_{\alpha_{j-1}} \supseteq D_y$, or $L_{j-1} \supseteq D_y$. It follows that

$$P[x_j \in D_y | x_1, \ldots, x_{j-1} \notin D_y] \;=\; \lambda(D_y)/\lambda(L_{j-1}) \;\geq\; \lambda(D_y)/\lambda(D)$$

Hence,

$$P[x_j \in D_y | x_1, \ldots, x_{j-1} \notin D_y] \;\geq\; \lambda(D_y)/\lambda(D)$$

as it is the average of the above term over all initial segments $x_1, \ldots, x_{j-1}$ of PLS sample paths with domain points not in $D_y$. Thus

$$P[N_{PLS}(z) > k-1] \;\leq\; [1 - \lambda(D_y)/\lambda(D)]^{k-1} \;=\; P[N_{PRS}(z) > k-1]$$

□

Notice that when $f \in L(M)$ with Lipschitz constant $M > 0$, two realisations of PLS immediately arise. The first is a stochastic analogue of Mladineo's algorithm, the second a stochastic analogue of the multidimensional bisection algorithm. In the former,

$$R_k = \bigcup_{i=1,\ldots,k} B_i$$

where $B_i$ is the ball of radius $(y_i - \alpha_k)/M$, centred at $x_i$. In the latter,

$$R_k = \bigcup_{i=1,\ldots,k} C_i$$

where $C_i$ is a standard simplex of radius $(y_i - \alpha_k)/M$, centred at $x_i$. The former is called *Spherical* PLS and the latter *Simplicial* PLS. These two special realisations

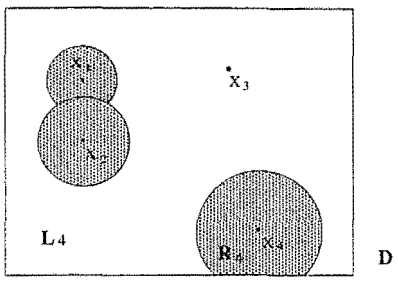



**Figure 5.2** Spherical PLS after four evaluations at $x_1, \ldots, x_4$, with $f(x_3) = \min \{f(x_1), \ldots, f(x_4)\}$. The shaded part is the removed region and the unshaded part the localisation $L_4$.

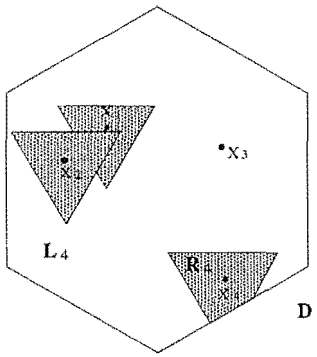



**Figure 5.3** Simplicial PLS after four evaluations at $x_1, \ldots, x_4$, with $f(x_3) = \min \{f(x_1), \ldots, f(x_4)\}$. The shaded part is the removed region and the unshaded part the localisation $L_4$.

of PLS are illustrated in Figures 5.2 and 5.3 for the case $d = 2$. Note that when $d = 1$ these two realisations of PLS reduce to the same algorithm.

In Chapter 2 the term "bracket" is used to describe the $d + 1$ dimensional region known to contain the global minimum point(s) $(x_*, y_*)$. This bracket can be used to implement PLS. The projection of this bracket onto the domain is what we term the localisation, see Figures 5.2 and 5.3. When $d = 1$, both Spherical and Simplicial PLS yield a bracket composed of disjoint similar triangles. For higher dimensions Simplicial PLS produces a union of overlapping, but similar, simplexes. For $d$-dimensional Spherical PLS the bracket is more complicated to describe.

We call both the above realisations, LPLS for "Lipschitz PLS". They are not trivial to implement. When $d = 1$ it is necessary to store a linked list of the intervals which comprise $L_k$. Selecting the next evaluation point is performed by choosing a random number in [0,1] and moving through the intervals to the $x_k$ value. Updating the localisation involves an updating of the linked list. For $d > 1$, Spherical PLS has the virtue of producing a tighter localisation than Simplicial PLS, since the removed ball always contains the removed simplex. On the other hand, choosing $x_k$ in Spherical PLS has so far been achieved through an acceptance-rejection approach, whereas with Simplicial PLS a linked list of simplex tops can be stored, and a procedure similar to the $d = 1$ case used to find $x_k$, see [78, 79].

For dimension $d > 1$, LPLS has not yet been satisfactorily implemented. This is because choosing a sample point uniformly on a general region is still an open problem. The localisation for Spherical PLS is a region formed by removing a collection of overlapping balls from $D$, while the localisation for Simplicial PLS consists of a set of overlapping standard simplexes. For the univariate case, however, LPLS has been coded and implemented. The following section compares the performances of different algorithms for a set of test functions.

Related work, in which sample points are chosen uniformly from the entire bracket, has recently appeared in [65].

## 5.4 Comparison of PRS, PAS, LPLS and the Piyavskii - Shubert algorithm in dimension one

In this section, we compare the performance of various algorithms on a set of univariate test functions. The algorithms we compare are PAS, PRS, Piyavskii-Shubert(PS) and LPLS. A `matlab` code was developed to carry out the implementation. PAS is implemented by running PRS and accepting only those points which provide im-

proved function evaluations. Generally speaking, compared with LPLS and PRS, Piyavskii-Shubert usually takes fewer function evaluations for the selected test functions. For functions with a large number of nearly equal global minima, however, LPLS can on average require less work than the Piyavskii-Shubert algorithm to attain modest accuracy. It is interesting to note that the work required by the Piyavskii-Shubert algorithm and its stochastic variant, LPLS, is very similar to that required by the theoretical PAS.

Two random selections of functions were made. We obtained the number of iterations until the global minimum was found to a specified tolerance, using the four algorithms.

The first selection consisted of 69 Lipschitz continuous functions on $[0,1]$ with M=1 and with the function values fixed at 0 at the end points. These test functions usually had a small number of local minima, generally one. These were produced by an easy modification of a procedure due to Graf, Mauldin and Williams described in [27, p.240-241].

To generate at random a Lipschitz continuous function $f$ on $[0,1]$ with M=1 and fixed function values $f(0) = f(1) = 0$, we first choose a function value at $x = 1/2$ according to a uniform distribution on $[-1/2, 1/2]$, which is the range in which $f$ can lie if $f \in L(1)$ and $f(0) = f(1) = 0$. Once the function value at $x = 1/2$ has been chosen, we choose a function value at $x = 1/4$ according to a uniform distribution on $[F_3(1/4), G_3(1/4)]$, where $F_3(x)$ is the tightest lower bounding function passing through points $(0,0), (1/2, f(1/2))$ with $M = 1$ defined in Chapter 3 and $G_3(x) = \min_{x\in[0,1]}\{f(x_i) + |x - x_i| : x_i = 0, 1/2, 1\}$, the tightest upper bounding function passing through points $(0,0), (1/2, f(1/2))$ and $(1,0)$ with $M = 1$. Independently, we choose a function value at $x = 3/4$ according to a uniform distribution on $[F_3(3/4), G_3(3/4)]$.

We continue this process by choosing function values at $x = 1/2^m, 3/2^m, \ldots,$ $(2^m - 1)/2^m$ according to the uniform distribution over $[F_k(x), G_k(x)]$, the interval between the tightest lower and upper bounding function values at $x$, where $F_k$ and $G_k$ are the tightest lower and upper bounding functions passing through previously determined points $(x_i, f(x_i))$. Terminate if $1/2^m$ is less than a given accuracy. Join each adjacent point generated as above by a line segment. This forms a function $f \in L(1)$. Figure 5.4 illustrates one such function.

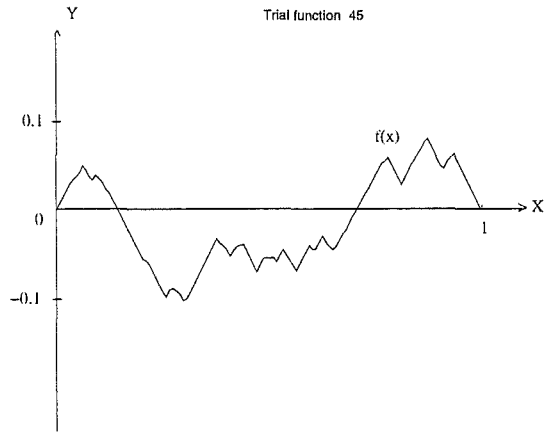



**Figure 5.4** A randomly generated Lipschitz function $f$ with $M = 1$ and $f(0) = f(1) = 0$, on the interval [0,1].

We initially chose 100 such randomly generated functions and dismissed the 31 which attained their global minimum at the end points. Table 5.1 compares the mean iteration numbers of the 69 test functions over 100 runs, for the four algorithms respectively. The termination criterion is chosen as the updated function minimisation within a distance of 0.0005 to the actual minimum of the test function.

Figure 5.5 gives the frequencies of the mean termination iteration numbers by drawing the histogram of the mean termination iteration numbers. It shows that for this class the algorithms ranked from best to worst are PAS, PS, LPLS and PRS.

| No. | PS | PAS | LPLS | PRS | No. | PS | PAS | LPLS | PRS |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 9 | 7.54 | 15.70 | 818.0 | 2 | 10 | 7.56 | 13.02 | 787.2 |
| 3 | 10 | 7.30 | 14.72 | 559.2 | 4 | 15 | 7.84 | 20.06 | 831.5 |
| 5 | 11 | 7.52 | 13.60 | 981.1 | 6 | 9 | 7.98 | 13.06 | 1060.9 |
| 7 | 10 | 7.58 | 13.50 | 564.8 | 8 | 12 | 8.18 | 16.42 | 937.0 |
| 9 | 10 | 7.32 | 15.08 | 920.2 | 10 | 12 | 7.98 | 18.48 | 941.4 |
| 11 | 10 | 8.06 | 17.18 | 931.3 | 12 | 17 | 7.76 | 20.06 | 861.2 |
| 13 | 8 | 7.90 | 13.96 | 985.7 | 14 | 11 | 7.46 | 14.86 | 789.8 |
| 15 | 14 | 9.18 | 15.68 | 999.5 | 16 | 10 | 6.86 | 12.08 | 411.2 |
| 17 | 5 | 7.84 | 10.36 | 1107.1 | 18 | 13 | 8.12 | 16.26 | 1032.5 |
| 19 | 14 | 7.98 | 17.44 | 588.8 | 20 | 11 | 8.54 | 13.96 | 818.1 |
| 21 | 10 | 7.36 | 14.06 | 608.7 | 22 | 12 | 7.44 | 19.10 | 814.7 |
| 23 | 9 | 8.14 | 11.40 | 1313.7 | 24 | 13 | 7.16 | 16.60 | 740.6 |
| 25 | 13 | 6.92 | 14.34 | 662.5 | 26 | 9 | 8.50 | 11.52 | 1136.9 |
| 27 | 10 | 7.44 | 11.64 | 496.3 | 28 | 20 | 6.84 | 18.20 | 333.1 |
| 29 | 10 | 7.40 | 12.92 | 803.0 | 30 | 10 | 7.80 | 15.34 | 1024.1 |
| 31 | 12 | 7.20 | 14.72 | 1047.5 | 32 | 8 | 5.92 | 10.94 | 93.3 |
| 33 | 15 | 6.76 | 18.68 | 414.7 | 34 | 10 | 6.02 | 14.46 | 106.5 |
| 35 | 10 | 8.94 | 16.76 | 875.0 | 36 | 13 | 8.18 | 15.54 | 992.1 |
| 37 | 12 | 7.48 | 21.14 | 747.9 | 38 | 10 | 7.76 | 15.26 | 825.3 |
| 39 | 9 | 8.10 | 12.06 | 772.1 | 40 | 10 | 7.66 | 15.64 | 985.7 |
| 41 | 16 | 8.18 | 17.98 | 980.4 | 42 | 15 | 7.56 | 15.62 | 512.5 |
| 43 | 15 | 7.96 | 19.64 | 787.2 | 44 | 13 | 7.30 | 14.96 | 814.6 |
| 45 | 10 | 6.42 | 18.06 | 375.1 | 46 | 9 | 7.40 | 12.02 | 776.9 |
| 47 | 13 | 6.10 | 14.34 | 123.7 | 48 | 15 | 7.60 | 20.40 | 763.0 |
| 49 | 10 | 8.04 | 15.84 | 656.3 | 50 | 20 | 7.80 | 23.20 | 953.0 |
| 51 | 9 | 7.20 | 16.46 | 596.0 | 52 | 12 | 7.28 | 14.12 | 1087.9 |
| 53 | 7 | 8.20 | 11.32 | 990.2 | 54 | 10 | 7.70 | 15.06 | 910.7 |
| 55 | 13 | 7.52 | 14.36 | 665.3 | 56 | 11 | 7.62 | 12.54 | 723.3 |
| 57 | 17 | 8.08 | 20.78 | 890.4 | 58 | 11 | 7.26 | 12.80 | 538.3 |
| 59 | 9 | 8.22 | 14.12 | 820.7 | 60 | 10 | 8.14 | 13.18 | 869.2 |
| 61 | 20 | 7.44 | 20.84 | 624.7 | 62 | 11 | 7.38 | 13.42 | 753.2 |
| 63 | 11 | 7.60 | 15.08 | 1004.5 | 64 | 10 | 8.30 | 14.90 | 625.1 |
| 65 | 9 | 7.60 | 14.72 | 618.3 | 66 | 13 | 6.52 | 12.92 | 198.7 |
| 67 | 13 | 6.94 | 18.26 | 632.1 | 68 | 7 | 8.12 | 11.38 | 807.0 |
| 69 | 13 | 7.46 | 16.32 | 899.1 | | | | | |

Table **5.1** Mean iteration numbers for the 69 test functions over 100 runs.

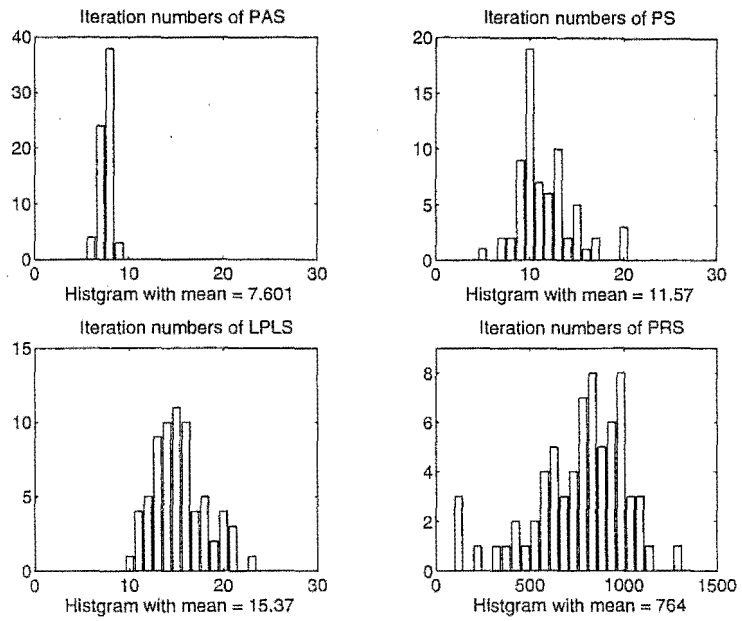



**Figure 5.5** Histograms showing the mean number of iterations to convergence, with $\epsilon = 0.0005$, for the 69 Graf-generated functions and the four algorithms. Note the different horizontal scale for PRS.

The second selection consisted of 50 Lipschitz continuous functions with M=1, of the form (1/A)sin(A$x$+B) where the A were randomly chosen on [5,50] and the B were randomly chosen on [0,1]. In this selection all of the functions have between one and eight global minima. Table 5.2 and 5.3 show that for this class the algorithms ranked from best to worst are PAS, LPLS, PS, and PRS for the modest accuracy $\epsilon = 0.1/A$, but PAS, PS, LPLS and PRS when the greater accuracy $\epsilon = 0.01/A$ is demanded.

In [12] Chuyan and Sukharev showed, under a mild condition, that results from adaptive stochastic global optimisation algorithms are in a certain sense no better than results from non-adaptive stochastic global optimisation algorithms. At first glance this appears to contradict the histograms shown for LPLS and PRS in Figure 5.5, since it is evident that the average behaviour of LPLS is far better than that for PRS. Chuyan and Sukharev, however, prove the equality of a "best worst-

| A | B | PS | PAS | LPLS | PRS | A | B | PS | PAS | LPLS | PRS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 14.853 | 0.047 | 5 | 3.30 | 5.20 | 9.74 | 35.549 | 0.679 | 3 | 3.08 | 5.93 | 6.68 |
| 47.061 | 0.384 | 1 | 2.94 | 5.80 | 7.59 | 28.374 | 0.831 | 8 | 3.07 | 6.08 | 6.52 |
| 6.556 | 0.054 | 5 | 3.08 | 4.50 | 7.12 | 28.837 | 0.671 | 3 | 2.73 | 5.75 | 7.60 |
| 5.346 | 0.383 | 4 | 2.90 | 3.77 | 6.71 | 8.008 | 0.418 | 1 | 3.31 | 4.55 | 8.19 |
| 35.905 | 0.589 | 3 | 2.91 | 5.71 | 7.46 | 46.870 | 0.846 | 4 | 3.15 | 7.11 | 7.71 |
| 28.712 | 0.092 | 12 | 2.96 | 6.15 | 8.14 | 34.426 | 0.416 | 1 | 3.37 | 5.87 | 7.48 |
| 36.554 | 0.910 | 14 | 2.93 | 5.17 | 7.78 | 39.299 | 0.263 | 5 | 2.94 | 6.36 | 8.05 |
| 7.136 | 0.736 | 1 | 3.21 | 4.97 | 7.48 | 19.771 | 0.633 | 9 | 2.83 | 5.38 | 6.97 |
| 39.039 | 0.991 | 5 | 3.21 | 5.68 | 8.38 | 21.440 | 0.247 | 1 | 3.16 | 5.49 | 7.27 |
| 49.215 | 0.723 | 19 | 3.05 | 5.74 | 6.96 | 38.901 | 0.652 | 5 | 3.16 | 5.23 | 7.06 |
| 8.271 | 0.632 | 1 | 2.99 | 5.22 | 9.53 | 44.812 | 0.273 | 7 | 2.86 | 5.85 | 7.54 |
| 24.639 | 0.767 | 11 | 2.92 | 5.42 | 7.40 | 26.498 | 0.238 | 9 | 3.05 | 4.94 | 6.30 |
| 17.371 | 0.359 | 5 | 2.87 | 4.76 | 6.90 | 12.493 | 0.487 | 7 | 3.11 | 4.95 | 5.98 |
| 45.395 | 0.909 | 1 | 2.83 | 6.00 | 6.54 | 7.725 | 0.905 | 1 | 2.92 | 4.80 | 9.10 |
| 27.704 | 0.516 | 8 | 2.96 | 5.27 | 7.64 | 19.357 | 0.987 | 1 | 2.96 | 6.12 | 7.96 |
| 27.229 | 0.266 | 8 | 3.28 | 5.53 | 8.14 | 9.083 | 0.948 | 7 | 3.27 | 5.30 | 10.12 |
| 8.319 | 0.501 | 1 | 2.94 | 5.30 | 8.96 | 22.286 | 0.277 | 1 | 3.14 | 5.71 | 8.06 |
| 46.122 | 0.530 | 1 | 2.94 | 6.69 | 8.22 | 25.900 | 0.941 | 7 | 3.08 | 5.73 | 6.50 |
| 7.254 | 0.762 | 1 | 2.96 | 4.46 | 7.63 | 39.659 | 0.828 | 5 | 3.11 | 6.38 | 6.38 |
| 10.641 | 0.016 | 3 | 3.46 | 5.89 | 11.54 | 35.981 | 0.868 | 8 | 2.97 | 5.43 | 5.90 |
| 33.330 | 0.736 | 1 | 2.88 | 5.85 | 7.63 | 37.644 | 0.999 | 14 | 2.84 | 5.89 | 6.30 |
| 44.986 | 0.233 | 7 | 2.93 | 5.45 | 7.49 | 18.785 | 0.351 | 8 | 2.79 | 5.19 | 6.37 |
| 28.097 | 0.591 | 9 | 3.07 | 5.59 | 8.58 | 43.069 | 0.412 | 5 | 2.71 | 5.98 | 6.62 |
| 42.868 | 0.269 | 5 | 3.08 | 4.51 | 7.11 | 23.693 | 0.537 | 10 | 2.96 | 4.47 | 6.73 |
| 26.056 | 0.287 | 11 | 3.12 | 5.42 | 7.10 | 13.025 | 0.154 | 7 | 3.12 | 5.05 | 6.61 |
| Overall | mean | 5.6 | 3.0 | 5.5 | 7.6 | | | | | | |

Table **5.2** Mean number of evaluations to convergence over 100 runs, using relative accuracy levels of $\epsilon = 0.1/A$, for the 50 sinusoidal functions.

| A | B | PS | PAS | LPLS | PRS | A | B | PS | PAS | LPLS | PRS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 14.853 | 0.047 | 9 | 4.41 | 8.78 | 24.03 | 35.549 | 0.679 | 3 | 3.87 | 12.21 | 20.53 |
| 47.061 | 0.384 | 19 | 4.10 | 13.48 | 20.95 | 28.374 | 0.831 | 14 | 4.25 | 12.33 | 25.05 |
| 6.556 | 0.054 | 5 | 4.12 | 7.77 | 23.94 | 28.837 | 0.671 | 14 | 4.17 | 11.72 | 24.08 |
| 5.346 | 0.383 | 4 | 3.99 | 6.83 | 16.24 | 8.008 | 0.418 | 7 | 4.16 | 8.52 | 26.41 |
| 35.905 | 0.589 | 19 | 3.87 | 12.06 | 23.39 | 46.870 | 0.846 | 19 | 4.23 | 14.54 | 23.53 |
| 28.712 | 0.092 | 15 | 4.52 | 12.17 | 25.95 | 34.426 | 0.416 | 17 | 4.17 | 11.76 | 20.46 |
| 36.554 | 0.910 | 19 | 4.40 | 12.73 | 22.69 | 39.299 | 0.263 | 19 | 3.94 | 12.09 | 20.28 |
| 7.1356 | 0.736 | 6 | 4.63 | 7.53 | 27.69 | 19.771 | 0.633 | 11 | 4.14 | 10.17 | 19.26 |
| 39.039 | 0.991 | 16 | 3.82 | 13.23 | 28.79 | 21.440 | 0.247 | 1 | 4.66 | 10.56 | 26.05 |
| 49.215 | 0.723 | 19 | 3.91 | 12.62 | 20.39 | 38.901 | 0.652 | 5 | 3.98 | 14.29 | 21.00 |
| 8.271 | 0.632 | 1 | 4.80 | 7.71 | 25.77 | 44.812 | 0.273 | 7 | 4.12 | 14.70 | 19.50 |
| 24.632 | 0.767 | 11 | 3.98 | 10.09 | 23.11 | 26.498 | 0.238 | 13 | 4.11 | 10.60 | 23.72 |
| 17.371 | 0.359 | 5 | 3.99 | 9.72 | 25.83 | 12.493 | 0.487 | 7 | 4.09 | 8.44 | 24.82 |
| 45.395 | 0.909 | 1 | 3.72 | 14.36 | 23.07 | 7.725 | 0.905 | 1 | 4.51 | 7.52 | 25.46 |
| 27.704 | 0.516 | 15 | 3.89 | 11.81 | 25.68 | 19.357 | 0.987 | 9 | 3.97 | 10.79 | 22.49 |
| 27.229 | 0.266 | 8 | 3.99 | 13.24 | 23.00 | 9.083 | 0.948 | 7 | 4.17 | 8.63 | 32.96 |
| 8.319 | 0.501 | 1 | 4.58 | 8.59 | 34.54 | 22.286 | 0.277 | 11 | 4.18 | 11.72 | 24.31 |
| 46.122 | 0.530 | 1 | 4.21 | 13.93 | 21.57 | 25.900 | 0.941 | 9 | 4.17 | 11.22 | 21.39 |
| 7.254 | 0.762 | 6 | 4.23 | 8.15 | 25.72 | 39.659 | 0.828 | 16 | 4.46 | 12.84 | 25.51 |
| 10.641 | 0.016 | 7 | 4.76 | 9.60 | 38.42 | 35.981 | 0.868 | 8 | 3.89 | 12.84 | 23.09 |
| 33.330 | 0.736 | 1 | 4.07 | 13.48 | 19.07 | 37.644 | 0.999 | 19 | 4.16 | 11.97 | 22.08 |
| 44.986 | 0.233 | 7 | 3.92 | 13.21 | 20.69 | 18.785 | 0.351 | 11 | 3.96 | 10.88 | 26.40 |
| 28.097 | 0.591 | 9 | 4.29 | 12.45 | 27.28 | 43.069 | 0.412 | 14 | 4.49 | 12.40 | 20.48 |
| 42.868 | 0.269 | 5 | 3.87 | 13.53 | 19.17 | 23.693 | 0.537 | 10 | 4.17 | 10.67 | 24.93 |
| 26.056 | 0.287 | 13 | 4.04 | 12.65 | 24.23 | 13.025 | 0.154 | 7 | 4.46 | 8.65 | 22.10 |
| Overall | mean | 9.6 | 4.2 | 11.2 | 23.9 | | | | | | |

Table **5.3** Mean number of evaluations to convergence over 100 runs, using relative accuracy levels of $\epsilon = 0.01/A$, for the 50 sinusoidal functions.

case" performance measure, for adaptive and non-adaptive algorithms. Figure 5.5 compares average behaviour, with objective functions drawn randomly using the Graf-Mauldin-Williams generation process. A much earlier paper, with results of the Chuyan and Sukharev type, dealing with deterministic algorithms and Lipschitz continuous functions, is that of Archetti and Betró [3].

## 5.5 Linking somewhat adaptive search and pure localisation search

Somewhat adaptive search was introduced to offer an efficiently implementable framework for attaining linear $\epsilon$-convergence complexity with dimension. Pure localisation search has been defined and implemented for LPLS for a set of test functions in Section 5.4. Does PLS ever achieve SAS under certain circumstances? This section is devoted to investigating this question.

In this section, we present a class of functions for which the LPLS algorithm achieves SAS. This links the ideal of SAS to the reality of PLS.

We now define a function of a single variable. For obvious reasons, it is called the (upside down) "witch's hat". For $h \in [0,1]$ the witch's hat of height $h$ is $w_h(x) = \min(|x|, h)$, for $x \in [-1,1]$. Note that $w_h$ is Lipschitzian function, with Lipschitz constant $M = 1$. That is, $w_h \in L(1)$ on the given domain. It will be convenient to call the graph of $w_h$ on $[-1,-h] \cup [h,1]$ the "brim", and $w_h$ on $[-h,h]$ the "cap". Figure 5.6 illustrates a function of this type.

The class of functions, $\mathcal{C}_h$, is defined as all those $f \in L(1)$ which agree with $w_h$ on $[-h,h]$ and elsewhere on $[-1,1]$ lie above $w_h$. When LPLS with $M = 1$ is applied to a function in $\mathcal{C}_h$, the localisation eventually becomes the best possible, the inverse image of all values less than or equal to the best known, that is, $f^{-1}(-\infty, \alpha_k]$. Note that for functions in $C_h$, when $\alpha_k \leq h$, this is just the interval $[-\alpha_k, \alpha_k]$. This

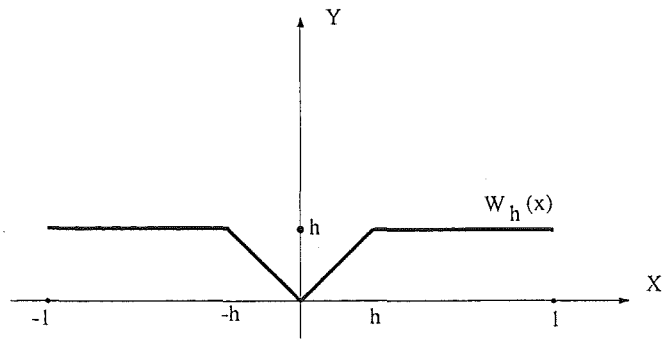



**Figure 5.6** A "witch's hat" function $w_h(x)$ with the "brim" on $[-1,-h],[h,1]$ and the "cap" on $[-h,h]$.

is formalised in the following theorem.

**Theorem 5.5.1** *Let $f$ be any function in $\mathcal{C}_h$, and for LPLS with M=1, let $N$ be the number of iterations until the localisation becomes the level set. Then*

$$E[N] \ < \ 6 + 26/h$$

The heart of the proof of Theorem 5.5.1 rests in recognizing that if we count $N_t$, the number of iterations until the lowest known evaluation is less than $t$, and also the number of subsequent iterations, $N_p$, until we can be sure that the localisation is the level set, then $N_t + N_p$ is greater than or equal to $N$ . Firstly, $E[N_t]$ can be proved to be less than $1/t$. We call an evaluation sequence $\{x_1, x_2, \ldots, x_k, \ldots\}$, generated by implementing LPLS once on a function $f \in \mathcal{C}_h$, a sample path. Then, the expectation number $E[N_p]$ is taken over all possible such sample paths after $N_t$ steps. We partition all such sample paths, and prove the probability of one step "success"(the localisation becomes the level set) is a positive number greater than or equal to $1/6$. Thus we can expect a finite number of steps to "success". Variable $N_p$ then has a negative binomial distribution, giving an upper bound on $E[N_p]$. A detailed proof is given in Appendix 1. We have as a corollary:

**Corollary 5.5.1** *For functions in $\mathcal{C}_h$, LPLS with $M = 1$ is SAS, with $\beta = 6+26/h$.*

Proof: Partition the sample paths as $\cup_{i=1}^{\infty}\Omega_i$, where $\Omega_i$ is the set of all sample paths for which the localisation becomes the level set at the $i$th iteration. Then for each $\omega \in \Omega_i$, $I_k(\omega)$, the number of iterations between the $(k-1)$st record and the $k$th record of $\omega$, is less than $i$ for all $k$, since for such sample paths there are at most $i$ iterations from any one record to the next. Thus

$$E(I_k) \leq \sum_{i=1}^{\infty} P[\Omega_i]\, i = E[N]$$

for any $k$. It follows that condition i) of SAS is satisfied with $\beta = 6 + 26/h$, using Theorem 5.5.1. Condition ii) of SAS follows immediately from Theorem 5.3.1. □

### Observed and theoretical results for LPLS on witch's hats in dimension one

Theorem 5.5.1 gave a theoretical upper bound on the average number of iterations until the localisation becomes the level set for the witch's hat. The following table compares this with the average observed number of iterations, over 1000 runs, thus showing that the theoretical bound is roughly ten times too large. The observed values of $E[N]$ are conservative estimates for $\beta$, as demonstrated in Corollary 5.5.1. Empirical tests have shown that $\beta$ is roughly one third of $E[N]$.

| $h$ | Observed | Theoretical |
|---|---|---|
| 1 | 4.8 | 20 |
| 1/2 | 7.4 | 46 |
| 1/3 | 9.8 | 72 |
| 1/4 | 12.1 | 108 |
| 1/8 | 21.4 | 202 |

Table **5.4** A comparison of the observed and the theoretical average number of iterations until the localisation becomes the level set, for the witch's hat with varying values of $h$.

### Observed results for Simplicial LPLS on witch's hat in higher dimensions

One higher dimensional analogue of the witch's hat is the upward facing simplicial cone defined over a simplicial domain. For this function with Simplicial PLS, the localisation, as in dimension one, becomes the level set.

| | Average observed no. iterations | |
|---|---|---|
| Dimension | Localisation = level set | Tolerance of 0.1 |
| 1 | 4.8 | 4.9 |
| 2 | 15 | 9.6 |
| 3 | 37 | 17.4 |
| 4 | – | 29.0 |

Table **5.5** The behaviour of Simplicial PLS on the higher dimensional analogue of the witch's hat.

The second column of Table 5.5 shows the average observed number of iterations, for 100 trials, for this to happen when the dimension is 1, 2 and 3. The third column of Table 5.5 shows the expected number of iterations for convergence, to within $\epsilon = 0.1$ of the global minimum, for Simplicial PLS for dimensions 1, 2, 3 and 4. Evidently these numbers are not linear in dimension. For PAS it is shown in [81] that this quantity is linear in dimension, so this implementation of LPLS is not "uniformly" close enough to PAS to maintain linearity in dimension. Empirically it appears that LPLS is SAS for the witch's hat in dimensions greater than one, but evidence suggests (Table 5.5) that the $\beta$ values are unbounded. If there were a bound, then LPLS would be linear in dimension (Theorem 5.2.1). However, if the $\beta$ values prove to be bounded by a function that is polynomial in dimension, then polynomial complexity of PLS would result.

CHAPTER 6

# Estimation of the Lipschitz constant of a function

## 6.1 Introduction

For global optimisation algorithms based on the Lipschitz continuity assumption of the objective function, it is critical to have an estimate of the Lipschitz constant. It is also necessary to have a good estimate, since the better the estimate the faster such algorithms converge, as described in [29]. In [58, 60], it has been shown that the local Lipschitz constant estimates are sufficient to guarantee theoretical convergence of a Lipschitz based algorithm and the numerical efficiency can be significantly enhanced by applying such estimates. It is obvious that a local Lipschitz constant is a "global" Lipschitz constant on a specific subset. In this chapter, we deal with the "global" Lipschitz constant estimation problem only.

Recall that if $M_0 > 0$ is a Lipschitz constant of a function $g$ over $D \subset \mathbf{R}^d$, the domain of $g$, then any $M \geq M_0$ is also a Lipschitz constant of $g$ over $D$. We say that $M_0$ is the least Lipschitz constant of $g$ over $D$, if for any $M < M_0$, $M$ is not a Lipschitz constant of $g$.

Finding the least Lipschitz constant of a function is itself a global optimisation problem. To verify this claim, suppose $g$ is a continuously differentiable function on a domain $D$, then the least Lipschitz constant will be $M = \max_{x \in D} || \bigtriangledown g(x)||$, the maximum magnitude of all directional derivatives of the function, since for any

$\epsilon > 0, M - \epsilon$ is not a Lipschitz constant of $g$, and $M + \epsilon$ is not a least Lipschitz constant of $g$. Notice that the above Lipschitz constant seeking global optimisation problem is a nonconvex problem. Suppose $g(x) = \sin x$, then $|| \bigtriangledown g(x)|| = |\cos x|$, a nonconvex function over any interval with length greater than $\pi$. We consider the problem of estimating the least Lipschitz constant of a function.

Existing methods dealing with the Lipschitz constant estimation problem in the literature fall into two categories. In the first the analytical form of the objective function and its derivatives are known explicitly, while in the second the form is unknown and only the function value can be evaluated. We term these objective functions *white box* and *black box* functions respectively.

For the white box problem, Shubert [69] gave a univariate example of Lipschitz constant estimation using the upper bound of the derivative. Mladineo [43] discussed the two dimensional case and chose the upper bound of $\sqrt{(\partial g/\partial x)^2 + (\partial g/\partial y)^2}$ as the estimate. Range inclusion techniques of interval analysis due to Moore [47] were used by Gourdin, Hansen and Jaumard [26] to estimate the Lipschitz constant for the likelihood function in the problem of maximum likelihood estimation of the three-parameter Weibull distribution. It is evident that any appropriate global optimisation method may be applied to find the Lipschitz constant of a white box objective function. If the estimation of the Lipschitz constant of $g$ requires the Lipschitz constant of the magnitude of the gradient of $g$, we find ourselves conserving the difficulty of the original optimisation problem.

On the other hand, for the black box problem, we have to find an upper bound for the magnitude of the gradient of the function using only the available function evaluations. Strongin proposed a method for univariate functions in [75]. After $k$ evaluations, the ordered evaluation points $x_1 < x_2 < \ldots < x_k$ and corresponding function values $g(x_1), g(x_2), \ldots, g(x_k)$ are available and an under-estimation of the

Lipschitz constant is given by $\hat{m} = \max_i\{|g(x_i) - g(x_{i-1})|/(x_i - x_{i-1})\}$. The Strongin estimate is then obtained by multiplying $\hat{m}$ by a factor $r > 1$. There is no guarantee, however, that the estimate $r\hat{m}$ is greater than or equal to the true Lipschitz constant. Hansen, Jaumard and Lu in [30] showed that no matter how a large factor $r$ is chosen, the Strongin estimator is an under-estimation of the true Lipschitz constant for a class of constructed Lipschitz functions, and hence Strongin's companion algorithm may terminate at a local optimum. In [18] de Haan proposed a method for estimating the minimum of a function using order statistics, and discussed necessary conditions on the objective function. While the method is similar in philosophy to our approach, our method requires only objective function evaluations, and not those of a derivative, to estimate the Lipschitz constant.

Our method builds on the ideas of Strongin and de Haan and addresses the Lipschitz constant estimation problem for a Lipschitz function alone. The development here can informally be described as follows. Recall that our aim is to find the supremum of all the slopes $|g(x) - g(y))|/||x - y||$ for distinct points $x$ and $y$ in $D$, the domain of $g$. If we sample $X$ and $Y$ uniformly on $D$, then the random variable $|g(X) - g(Y)|/||X - Y||$ itself has cumulative distribution function $F$, which we term the *slope distribution* of $g$. The upper bound of its support is the Lipschitz constant we need. Unfortunately we do not know the form of $F$ for an arbitrary objective function $g$.

Suppose now that we draw a random sample of say $n$ absolute slopes, and consider the distribution of the largest. Provided that $F$ satisfies the Gnedenko condition, given in the next section, then the distribution of the largest absolute slope is known to be approximately Reverse Weibull. Its location parameter (the upper bound of the support) will estimate our Lipschitz constant.

We formalise these ideas in Section 6.2 and illustrate the method with numerical results in Section 6.3. In Section 6.4 we show that for a large class of univariate

functions the slope distribution does satisfy the Gnedenko condition. The multivariate situation is also investigated in this section. The conclusion is contained and directions for future research are discussed in Section 6.5.

Much of the material in this chapter is to appear in [80].

## 6.2 The method

Let $D$ be a compact convex subset of $\mathbf{R}^d$ and $L(M)$ be all Lipschitz functions $g$ on $D$ with Lipschitz constant $M$.

We begin by illustrating the method with a simple example. In Figure 6.1 we show a univariate function $g(x) = x - x^3/2$ on $[-1, 1]$. The Lipschitz constant $M$ equals one, since $g'(0) = \max_{x\in[-1,1]} g'(x) = 1$. We choose $x$ and $y$ uniformly and independently in $[-1, 1]$. In the example $x = -0.78$ and $y = 0.84$. The absolute value of the slope estimate is $s = |g(y) - g(x)|/|y - x| = 0.67$.

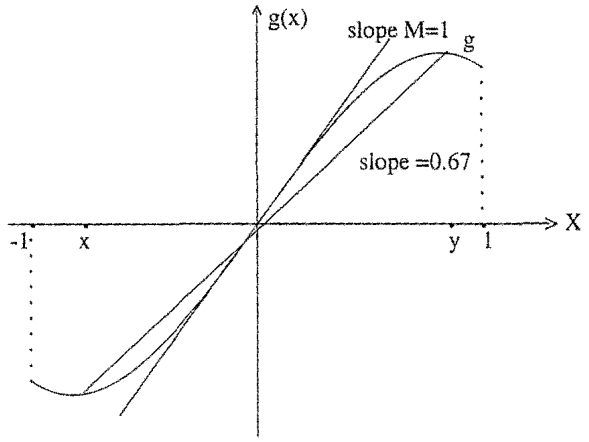



**Figure 6.1** The objective function $g$ and a sampled slope of 0.67 .

Suppose now that we repeat this procedure many times. Then the cumulative distribution of such slopes absolute values converges to the cumulative slope distribution $F$, shown in Figure 6.2. Note that the associated probability density function will have support with upper limit $M$.

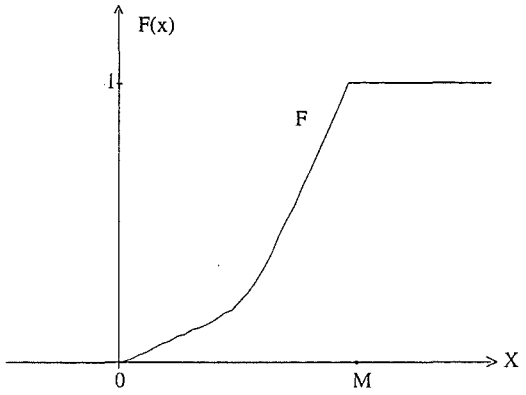



**Figure 6.2** The cumulative distribution function for absolute values of slopes.

Now take a random sample of size $n = 5$ from this distribution. Let $l$ be the largest of these absolute slopes. Then the cdf of $l$ will be $F^5(x)$. This is easy to check: let $s_1, \ldots, s_5$ be identically distributed independent random variables with cumulative distribution function $F(x)$, then $F_l(x) = \text{Prob}[\max\{s_1, \ldots, s_5\} \leq x] = \text{Prob}[s_1 \leq x, s_2 \leq x, \ldots, s_5 \leq x] = \text{Prob}[s_1 \leq x]\text{Prob}[s_2 \leq x] \ldots \text{Prob}[s_5 \leq x] = F^5(x)$. This maximum slope cumulative distribution function is shown in Figure 6.3. Note that the probability density function of $l$ will have the same support upper bound $M$.

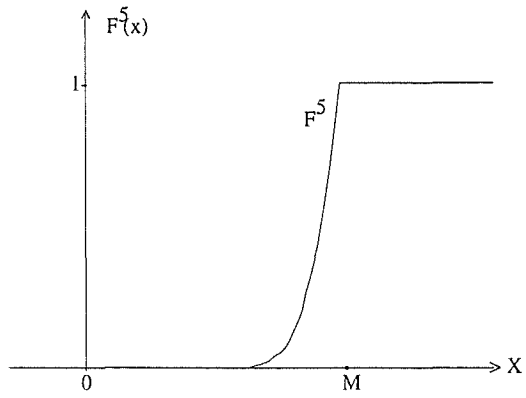



**Figure 6.3** The cumulative distribution function for the largest of a sample of five absolute slopes.

In general, what is the distributional form of $F^n$? If we know the distribution towards which $F^n$ tends as $n$ increases, then the estimate of the location parameter

M can be used as an estimate of the Lipschitz constant of $g$.

The theory of extreme value distributions tells us that there are just three distributional types for extreme values. Since the support of $F$ is bounded above, standard theory yields that $F^n$ is approximately Reverse Weibull, provided the original distribution $F$ satisfies the Gnedenko condition

$$\lim_{\epsilon \to 0^+} \frac{1 - F(M - c\epsilon)}{1 - F(M - \epsilon)} = c^k.$$

for any $c > 0$ and some constant $k > 0$. This result first appeared in [25] .

The cumulative distribution function of the three-parameter Reverse Weibull distribution is

$$H(l) = \begin{cases} \exp\{(-(u-l)^w)/v\} & \text{if} \quad l \leq u \\ 1 & \text{if} \quad l > u, \end{cases}$$

where $u \in \mathbf{R}$ is the location parameter, $v > 0$ is the scale parameter and $w > 0$ is the shape parameter. A precise description of the manner in which $F^n$ converges to a Reverse Weibull distribution is given in Section 6.4. We fit a Reverse Weibull distribution to a sample of largest absolute slopes and use the estimate of the location parameter as an estimate of $M$.

We now formally present a stochastic procedure to estimate the Lipschitz constant $M$ of $g$. Our method assumes only that the function values can be evaluated. Larger values of the absolute slope will be found for $(x, y)$ pairs chosen in the region of $D \times D$ where $x$ and $y$ are close. For this reason we set up a sampling scheme in Step 1 which allows pairs to be chosen according to a uniform distribution on a region around the "diagonal", where $x = y$.

***Step 1: Sample the slopes*** *Given $\delta > 0$, choose pairs $(x_i, y_i)$ uniformly on $\{(x, y) = (x(1), \ldots, x(d), y(1), \ldots, y(d)) \in D \times D : |x(k) - y(k)| \leq \delta$ for $k = 1, \ldots, d\}$ and evaluate*

$$s_i = \frac{|g(x_i) - g(y_i)|}{||x_i - y_i||}$$

*for* $i = 1, \ldots, n$.

***Step 2: Calculate the maximum slope*** *Let*

$$l = \max\{s_1, \ldots, s_n\}.$$

*Steps 1 and 2 are performed m times, giving* $l_1, \ldots, l_m$.

***Step 3: Fit the Reverse Weibull*** *Fit a three-parameter Reverse Weibull distribution to* $l_1, \ldots, l_m$.

***Output:*** *Our estimate* $\hat{M}$, *of* $M$, *is the location parameter of the fitted Reverse Weibull distribution.*

We remark that we have converted the Lipschitz constant estimation problem into a routine curve fitting problem. Note that a nonlinear curve fitting problem, in general, is itself a global optimisation problem. For our application here we can reduce a general Lipchitz constant estimation problem, a global optimisation problem, to a specified Weibull probability density function fitting problem. This fitting problem can be solved by implementing an algorithm which will be described in the next section.

## 6.3 Reverse Weibull fitting and numerical results

Does the method give good results? To investigate this question, we begin by observing that if a random variable has a Reverse Weibull distribution, then its negative has a Weibull distribution. Thus we can employ standard Weibull fitting methods.

Methods for finding maximum likelihood estimates for the three-parameter Weibull distribution are discussed in detail in the surveys in [48] and [84]. A detailed discussion is also presented in the context of global optimisation in [87].

We use a combination of profile likelihood and the method of moments to fit the Weibull distribution. For fixed $u$, we can straightforwardly use the first and

second moments to find $v$ and $w$. We then select the $(u, v, w)$ combination which maximises the likelihood of the observed maximum slopes. After choosing $nm$ pairs $(x_{ij}, y_{ij})$ uniformly on $\{(x, y) \in D \times D : |x(k) - y(k)| \leq \delta \text{ for } k = 1, \ldots, d\}$ for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, we have samples of largest slopes :

$$l_j = \max_i \frac{|g(x_{ij}) - g(y_{ij})|}{||x_{ij} - y_{ij}||}, \quad j = 1, \ldots, m.$$

Then $t_j = U_0 - l_j$ can be treated as an observation of a Weibull random variable $T$, where $U_0$ is a given upper bound for absolute slopes. The density function of the Weibull distribution is

$$f(t; u, v, w) = \begin{cases} \frac{w}{v}(t-u)^{w-1} \exp\left(-\frac{(t-u)^w}{v}\right) & \text{if} \quad t \geq u \\ 0 & \text{if} \quad t < u, \end{cases}$$

We estimate the parameters $u, v$ and $w$ of the Weibull distribution by maximising the log-likelihood function:

$$\begin{aligned} L(u, v, w) &= \sum_{j=1}^{m} \log f(t_j; u, v, w) \\ &= m \log(w/v) - (1/v) \sum_{j=1}^{m} (t_j - u)^w + (w-1) \sum_{j=1}^{m} \log(t_j - u), \end{aligned}$$

subject to the moment constraints

$$\bar{t} - u = v^{1/w} \Gamma(1 + 1/w)$$

$$s_t^2 = v^{2/w}(\Gamma(1 + 2/w) - \Gamma^2(1 + 1/w))$$

$$0 \leq u \leq \min t_j, \quad v > 0, \quad w > 0$$

where $\bar{t}$ is the mean of the observations, $s_t$ is the standard deviation of the observations and $\Gamma$ is the Gamma function.

The above constraints arise from the moment identities. The first and second order moments of $T$ about zero are

$$E(T) = \int_u^\infty t f(t; u, v, w) dt = u + v^{1/w} \Gamma(1 + 1/w) \quad \text{and}$$

$$E(T^2) = \int_u^\infty t^2 f(t;u,v,w)dt = u^2 + 2uv^{1/w}\Gamma(1+1/w) + v^{2/w}\Gamma(1+2/w)$$

respectively.

We equate the observed first and second central moments to their values expressed in terms of $u, v$ and $w$. Notice that

$$\frac{1}{m}\sum_{j=1}^m t_j^2 = s_t^2 + \bar{t}^2 \quad \text{and} \quad u^2 + 2uv^{1/w}\Gamma(1+1/w) = \bar{t}^2,$$

so we obtain the above constraints.

For fixed $u$, the parameters $v$ and $w$ can be estimated through the above moment method, therefore the likelihood function becomes a function of $u$ only, denoted $L^{**}(u)$. We then maximise $L^{**}(u)$ by using the grid method. If we estimate $u$ using the above procedure, then $U_0 - u$ becomes an estimate of the least upper bound of the largest slope $l$ and can be chosen as the Lipschitz constant estimate.

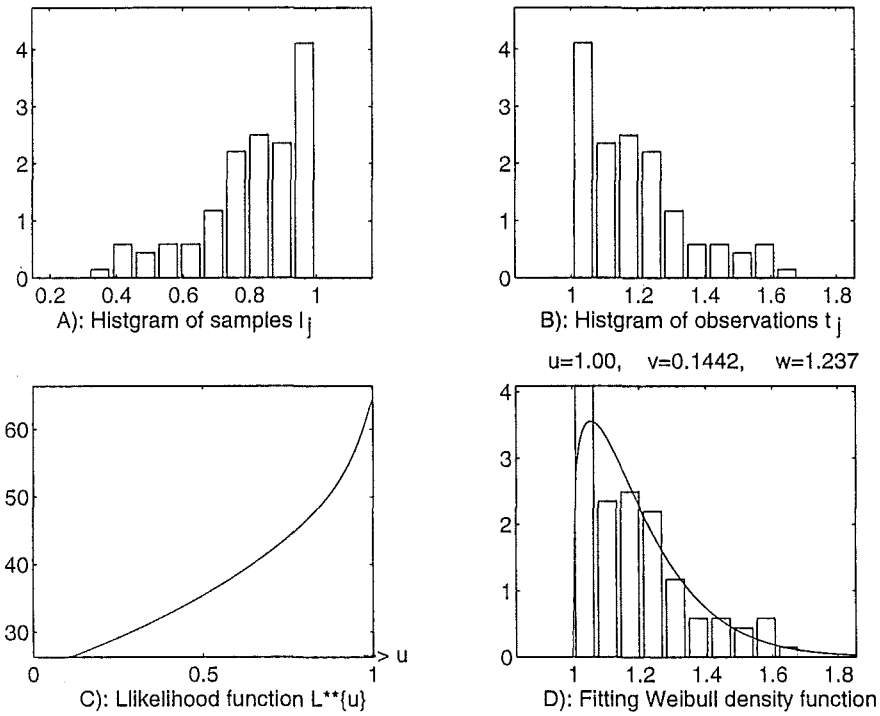



**Figure 6.4** Illustration of the Lipschitz constant estimation method for function $g(x) = \sin x$ on $[0, 2\pi]$ with $n = 5$, $m = 100$ and $U_0 = 2$.

A typical fitting procedure is illustrated in Figure 6.4 for the function $g(x) = \sin x$ on $[0, 2\pi]$, $n = 5$, $m = 100$ and $U_0 = 2$. Graphic A shows the histogram of maximum absolute slopes $l_j$ of 100 samples of 5 pairs. Graphic B is the histogram of reflected-and-translated maximum absolute slopes, the histogram of observations $t_j = U_0 - l_j$. Graphic C shows the graph of the function $L^{**}(u)$. Graphic D shows the Weibull density curve fitted to the samples $t_j$ with the list of estimates $u, v$ and $w$. The estimated Lipschitz constant of $g$ is then $U_0 - u = 2 - 1.00 = 1.00$.

Gourdin, Hansen and Jaumard in [26] proposed a global algorithm to solve this very problem. We used their method on occasions to check our results.

### Objective functions of one variable

For the univariate case, we describe results on four test functions. The first two test function are the simple functions $x - x^3/3$ on $[-1, 1]$ and $\sin x$ on $[0, 2\pi]$ and $[0, 5\pi]$. Both have Lipschitz constant $M = 1$. The other two test functions are drawn from [77, p.177]: $\sin x + \sin(2x/3)$ on $[3.1, 20.4]$ with $M = 1.67$ and $-\sum_{k=1}^{5} \sin((k+1)x+k)$ on $[-10, 10]$ with $M = 67$. Tables 6.1, 6.2, 6.3 and 6.4 give the estimates of $M$ for various choices of $n$ and $m$ with the sampling performed according to a uniform distribution on $[a, b] \times [a, b]$.

| | $m = 25$ | $m = 50$ | $m = 75$ | $m = 100$ |
|---|---|---|---|---|
| $n = 3$ | 0.9990±0.0074 | 1.0020±0.0042 | 1.0000±0.0000 | 1.0000±0.0000 |
| $n = 5$ | 1.0000±0.0067 | 1.0000±0.0000 | 1.0000±0.0000 | 1.0000±0.0000 |
| $n = 7$ | 1.0000±0.0000 | 1.0000±0.0000 | 1.0000±0.0000 | 1.0000±0.0000 |

Table **6.1** Lipschitz constant estimates for $x - x^3/3$ on $[-1, 1]$, using uniform sampling on $[-1, 1] \times [-1, 1]$. Here $M = 1$.

| $[a,b]$ | | $m=25$ | $m=50$ | $m=75$ | $m=100$ |
|---|---|---|---|---|---|
| | $n=3$ | 1.0130±0.0540 | 0.9800±0.0183 | 0.9840±0.0178 | 0.9910±0.0032 |
| $[0,2\pi]$ | $n=5$ | 0.9930±0.0164 | 0.9940±0.0052 | 1.0010±0.0047 | 1.0000±0.0047 |
| | $n=7$ | 0.9920±0.0230 | 1.0000±0.0000 | 1.0010±0.0032 | 1.0010±0.0032 |
| | $n=3$ | 1.0790±0.2347 | 1.0040±0.0534 | 1.0270±0.0427 | 0.9720±0.0063 |
| $[0,5\pi]$ | $n=5$ | 0.9820±0.0326 | 1.0020±0.0155 | 1.0090±0.0152 | 0.9980±0.0114 |
| | $n=7$ | 0.9830±0.0221 | 1.0000±0.0047 | 1.0100±0.0141 | 1.0050±0.0053 |

Table **6.2** Lipschitz constant estimates for $\sin x$ on $[0,2\pi]$ and $[0,5\pi]$, using uniform sampling on $[a,b]\times[a,b]$. Here $M=1$.

| | $m=25$ | $m=50$ | $m=75$ | $m=100$ |
|---|---|---|---|---|
| $n=3$ | 2.4120±0.7605 | 2.5840±0.6702 | 2.8590±0.4459 | 2.0090±0.6839 |
| $n=5$ | 2.4060±0.7327 | 2.4760±0.6712 | 2.6170±0.5781 | 2.0170±0.6784 |
| $n=7$ | 1.9520±0.5713 | 1.9920±0.5289 | 1.8270±0.2957 | 1.7140±0.2049 |
| $n=9$ | 1.5840±0.1327 | 1.7220±0.1078 | 1.7120±0.0863 | 1.7190±0.0926 |
| $n=11$ | 1.7000±0.1441 | 1.6750±0.1918 | 1.6720±0.0379 | 1.6740±0.0398 |

Table **6.3** Lipschitz constant estimates for $\sin x+\sin(2x/3)$ on $[3.1,20.4]$, using uniform sampling on $[3.1,20.4]\times[3.1,20.4]$. Here $M=1.67$.

| | $m=25$ | $m=50$ | $m=75$ | $m=100$ |
|---|---|---|---|---|
| $n=3$ | 47.3820±24.5935 | 54.3096±16.0915 | 61.0462±10.3427 | 60.1370±9.9099 |
| $n=5$ | 59.4610±13.7087 | 60.0020±12.0155 | 58.3790±10.7152 | 59.7570±9.1609 |
| $n=7$ | 61.5820±6.2873 | 55.4950± 5.1661 | 62.3840±8.0365 | 67.7140±6.5446 |
| $n=9$ | 62.2960±10.6238 | 60.2250± 9.8699 | 66.4400±7.6948 | 69.4510±6.3622 |
| $n=11$ | 60.1370±9.9099 | 63.6500± 8.6893 | 68.7500±6.3424 | 72.0440±5.0883 |

Table **6.4** Lipschitz constant estimates for $-\sum_{k=1}^{5}\sin((k+1)x+k)$ on $[-10,10]$, using uniform sampling on $[-10,10]\times[-10,10]$. Here $M=67$.

Table 6.5 gives estimates of $M$ for three of these functions using $m = 100$, various choices of $n$ and with $\delta = 0.05$. In the next section we show that the slope distribution of these test functions satisfies the Gnedenko condition. All of the estimates are given in the form $\alpha \pm \beta$, where $\alpha$ is the mean and $\beta$ the standard deviation over ten runs for the given $n$ and $m$.

| Function | | $x - x^3/3$ | $\sin x + \sin(2x/3)$ | $-\sum_{k=1}^{5} \sin((k+1)x + k)$ |
|---|---|---|---|---|
| Interval | | $[-1, 1]$ | $[3.1, 20.4]$ | $[-10, 10]$ |
| $n = 3$ | $m = 100$ | 1.0000±0.0000 | 1.7040±0.0227 | 73.3870±1.8872 |
| $n = 5$ | $m = 100$ | 1.0000±0.0000 | 1.6790±0.0074 | 68.4040±0.0975 |
| $n = 7$ | $m = 100$ | 1.0000±0.0000 | 1.6750±0.0085 | 68.4250±0.0474 |
| $n = 9$ | $m = 100$ | 1.0000±0.0000 | 1.6720±0.0042 | 68.4080±0.0282 |

Table **6.5** Lipschitz constant estimates for the three test functions using $\delta = 0.05$.

Note that the estimates improve as $n$ and $m$ increase. Note also that use of a small $\delta$ improves these results, especially when the domain interval is large. This is due to the fact that when the interval is large, sampling from $[a, b] \times [a, b]$ rarely produces a pair $(x, y)$ in the region where the maximum absolute slope is achieved.

The following example allows us to compare of our numerical result to Strongin's method. The test function is a modification of that introduced in [30] by Hansen, Jaumard and Lu Let

$$g(x) = \begin{cases} \max(2\sin(\frac{\beta\pi x}{2}), x) & \text{if} \quad 0 \leq x < 2/\beta \\ x & \text{if} \quad 2/\beta \leq x \leq 1 \end{cases}$$

where $\beta = 2(\frac{2r}{r+1})^{\Lambda}$, $\Lambda = \text{ceil}\{\frac{\ln r}{\ln 2r - \ln(r+1)}\} - 2$, $r$ is the multiplier of Strongin's algorithm and ceil $(x)$ is the smallest integer greater than or equal to $x$. Figure 6.5 shows $g$ for the case where $r = 2$.

It can be proved that for any multiplier $r > 1$, Strongin's estimate of the Lipschitz constant is always $r$, but the true Lipschitz constant $M$ is $\pi\beta$, which equals

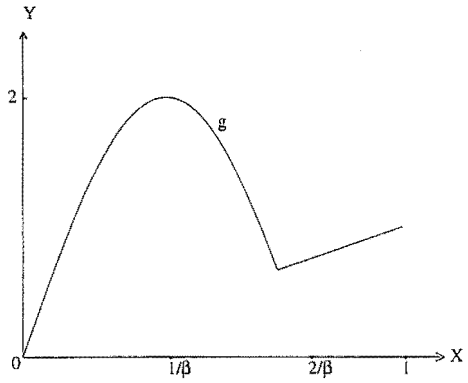



**Figure 6.5** Hansen-Jaumard-Lu test function with $r = 2$.

$2\pi(\frac{2r}{r+1})$, if $1 < r \leq 4$, as $\Lambda = 1$. Thus the Strongin estimate is an under-estimate and hence his algorithm converges to a non-global local optimum. Table 6.6 compares Lipschitz constant estimates using the Strongin method (S) and the Reverse Weibull method (RW) for two such Hansen-Jaumard-Lu test functions on $[0, 1]$.

| Factor $r$ | True $M$ | S estimate | $n$ | $m$ | RW estimate $\delta > \lvert b-a \rvert$ | RW estimate $\delta = 0.05$ |
|---|---|---|---|---|---|---|
| | | | 3 | 100 | $10.1394 \pm 1.6007$ | $8.3052 \pm 0.0253$ |
| $r = 2$ | $\pi\beta$ | | 5 | 100 | $8.5384 \pm 0.2736$ | $8.3792 \pm 0.0042$ |
| | $= 8.3776$ | 2 | 7 | 100 | $8.4334 \pm 0.1758$ | $8.3752 \pm 0.0042$ |
| | | | 9 | 100 | $8.3644 \pm 0.0585$ | $8.3742 \pm 0.0095$ |
| | | | 20 | 100 | $8.3802 \pm 0.0127$ | $8.3772 \pm 0.0000$ |
| | | | 3 | 100 | $13.1292 \pm 1.7332$ | $9.3182 \pm 0.0348$ |
| $r = 3$ | $\pi\beta$ | | 5 | 100 | $9.8362 \pm 0.5191$ | $9.4192 \pm 0.0042$ |
| | $= 9.4248$ | 3 | 7 | 100 | $9.6892 \pm 0.2656$ | $9.4222 \pm 0.0053$ |
| | | | 9 | 100 | $9.3772 \pm 0.1124$ | $9.4202 \pm 0.0095$ |
| | | | 20 | 100 | $9.4249 \pm 0.0495$ | $9.4262 \pm 0.0057$ |

Table **6.6** Lipschitz constant estimation comparison: the Strongin estimate and the Reverse Weibull estimates for a large $\delta$ value and a small $\delta$ value.

### Objective functions of many variables

We now consider objective functions defined on a domain $D$ in $\mathbf{R}^d$ for $d \geq 2$. For such multivariate objective functions, we first list the results for linear functions. Some partial results concerning the Gnedenko condition in this case will be developed in the next section. Table 6.7 gives the results of estimations of Lipschitz constants for $g(x) = (1/\sqrt{d})(x_1 + x_2 + \ldots + x_d)$. The domain is the $d$-cube $\{x = (x_1, x_2, \ldots, x_d) : -1 \leq x_i \leq 1 \text{ for } 1 \leq i \leq d\}$. That the Gnedenko condition holds in a certain sense for such a function will be shown later. Despite the simplicity of $g$, this task is challenging.

| Dimension $d$ | True $M$ | $n$ | $m$ | Estimate $\delta > 2$ | Estimate $\delta = 0.05$ |
|---|---|---|---|---|---|
| 2 | 1 | 3 | 100 | 1.0000 ± 0.0000 | 1.0000 ± 0.0000 |
| 2 | 1 | 5 | 100 | 1.0000 ± 0.0000 | 1.0000 ± 0.0000 |
| 2 | 1 | 10 | 100 | 1.0000 ± 0.0000 | 1.0000 ± 0.0000 |
| 3 | 1 | 3 | 100 | 1.0010 ± 0.0032 | 1.0000 ± 0.0000 |
| 3 | 1 | 5 | 100 | 1.0000 ± 0.0000 | 1.0000 ± 0.0000 |
| 3 | 1 | 10 | 100 | 1.0000 ± 0.0000 | 1.0000 ± 0.0000 |
| 5 | 1 | 3 | 100 | 0.9660 ± 0.0310 | 1.0360 ± 0.0566 |
| 5 | 1 | 5 | 100 | 0.9720 ± 0.0193 | 0.9930 ± 0.0195 |
| 5 | 1 | 10 | 100 | 0.9870 ± 0.0157 | 0.9930 ± 0.0106 |
| 10 | 1 | 3 | 100 | 0.8840 ± 0.1839 | 1.2370 ± 0.2543 |
| 10 | 1 | 5 | 100 | 0.9090 ± 0.1312 | 1.0990 ± 0.1863 |
| 10 | 1 | 10 | 100 | 0.9080 ± 0.0951 | 1.0510 ± 0.1634 |

Table **6.7** Lipschitz constant estimates for linear functions $g$ for a range of dimensions and sample sizes $n$ and $m$. Here $g(x_1, x_2, \cdots, x_d) = \frac{1}{\sqrt{d}}(x_1 + x_2 + \cdots + x_d)$.

Our next example shows numerical results for the function

$$g(x_1, x_2) = (1/\sqrt{2})(x_1 + x_2 - (1/3)(x_1^3 + x_2^3))$$

on $[-1, 1] \times [-1, 1]$. Results are shown in Table 6.8.

| True $M$ | $n$ | $m$ | Estimation $\delta > 2$ | Estimation $\delta = 0.05$ |
|---|---|---|---|---|
| 1 | 3 | 100 | $0.9590 \pm 0.0191$ | $0.9800 \pm 0.0141$ |
| 1 | 5 | 100 | $0.9770 \pm 0.0149$ | $1.0060 \pm 0.0255$ |
| 1 | 10 | 100 | $0.9890 \pm 0.0110$ | $1.0010 \pm 0.0074$ |

Table **6.8** Lipschitz constant estimates for $g(x_1, x_2) = (1/\sqrt{2})(x_1 + x_2 - (1/3)(x_1^3 + x_2^3))$.

The above numerical examples show that the RW method works well for a class of univariate functions. The results for linear functions of more than one variable grow less accurate as the dimension goes up, as revealed by Table 6.5. Notice that there are two barriers to success: the first is that our samples size $n$ must be finite (usually not very large) and so we have to replace the limiting Reverse Weibull by an approximation. The second is that the probability of obtaining an absolute slope close to the maximum in a higher dimension, through uniform selection of a sample pair in the domain, is much less than in a lower dimension. The above reasons make estimation more difficult in higher dimensions.

## 6.4 The Gnedenko condition

In order that the Reverse Weibull distribution should approximate the distribution of the maximum absolute slope, a condition must be satisfied by the original cumulative distribution function $F$ : it must satisfy the Gnedenko condition. Our aim is to find

conditions on the objective function $g$ which will ensure that $F$ will satisfy the Gnedenko condition.

## 1. Preliminary results

The following proposition formally presents the necessary and sufficient Gnedenko condition on $F$.

**Proposition 6.4.1** *Let $M = sup\{s : F(s) < 1\}$ be finite and $L$ be the largest value in a random sample of size $n$ drawn from $F$. Then there are sequences $a_n$ and $b_n > 0$ such that $F^n(a_n + b_n l)$ converges pointwise to the standard Reverse Weibull distribution $H(l; k)$ if and only if for any $c > 0$,*

$$\lim_{\epsilon \to 0^+} \frac{1 - F(M - c\epsilon)}{1 - F(M - \epsilon)} = c^k,$$

*the Gnedenko condition. Here $H(l; k)$ is the Reverse Weibull distribution with $u = 0$, $v = 1$ and $w = k$.*

The proof of this proposition can be found in [21, p.53-57 and pp.87-91]. The standard Reverse Weibull distribution involves only the shape parameter $w = k > 0$. See also [50], for work related to the concave minimisation problem and [87] describing additional work on this subject.

The above result tells us that an affine transformation, $a_n + b_n l$, of our absolute maximum slopes has an approximate Reverse Weibull distribution. Thus the maximum slope itself has an approximate Reverse Weibull distribution. For completeness this is proved in the following lemma.

**Lemma 6.4.1** *Suppose that the cumulative distribution function of a random variable $\xi$ is Reverse Weibull, then the cumulative distribution function of $\eta = a + b\xi$ for $b > 0$ is also Reverse Weibull.*

Proof: The cumulative distribution function of $\xi$ is

$$F_\xi(l) = \begin{cases} \exp(\frac{-(u-l)^w}{v}) & \text{if } l \leq u \\ 1 & \text{if } l > u \end{cases}$$

with $u \in \mathbf{R}$, $v > 0, w > 0$. Then the cumulative distribution of $\eta$ is

$$\begin{aligned} & F_\eta(l) = \text{Prob}(\eta < l) \\ = \ & \text{Prob}(a + b\xi < l) \\ = \ & \text{Prob}(\xi < \frac{l-a}{b}) \quad (\text{since} \quad b > 0) \\ = \ & F_\xi(\frac{l-a}{b}) \\ = \ & \begin{cases} \exp(\frac{-(a+bu-l)^w}{b^w v}) & \text{if } l \leq a + bu \\ 1 & \text{if } l > a + bu \end{cases} \\ = \ & \begin{cases} \exp(\frac{-(u'-l)^{w'}}{v'}) & \text{if } l \leq u' \\ 1 & \text{if } l > u' \end{cases} \end{aligned}$$

with $u' = a + bu$, $v' = b^w v > 0$ and $w' = w > 0$ in $H(l)$. □

To summarise, the maximum absolute slope has approximate Reverse Weibull distribution: $F^n(l)$ is approximately $H((l - a_n)/b_n; k)$.

To ensure that the absolute slope distribution $F$ satisfies the Gnedenko condition, we must consider the absolute slope function, defined for $(x, y) \in D \times D$, by

$$s(x, y) = |g(x) - g(y)|/||x - y|| \quad \text{for} \quad x \neq y.$$

Since the measure of $\{(x, y) \in D \times D : x = y\}$ is 0 in $D \times D$ the exclusion of this definition for $s$ on $\{(x, y) \in D \times D : x = y\}$ does not affect our later discussion in which we use the measure of a subset of $D \times D$.

For fixed $\epsilon > 0$ and $c > 0$ let

$$D_\epsilon = \{(x, y) \in D \times D : s(x, y) \geq M - \epsilon\} \quad \text{and}$$

$$D_{c\epsilon} = \{(x, y) \in D \times D : s(x, y) \geq M - c\epsilon\}.$$

Typical $D_\epsilon$ and $D_{c\epsilon}$ for the case $d = 1$ are illustrated in Figure 6.6, with $c > 1$.

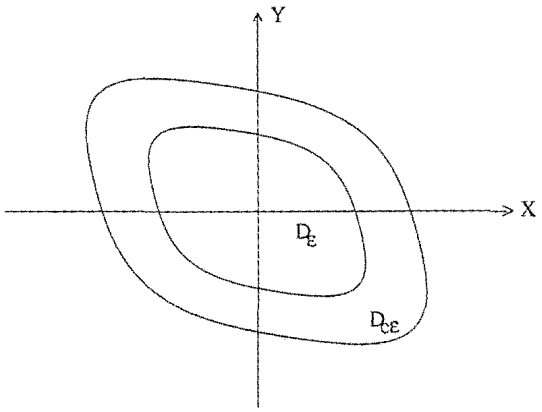



**Figure 6.6** Level sets $D_\epsilon$ and $D_{c\epsilon}$ for the absolute slope function $s$ of the objective function $g$.

Recall that the slope distribution function $F$ is given by the relation

$$F(t) = \text{Prob}\{(x, y) \in D \times D : s(x, y) \leq t\} = \mu\{(x, y) \in D \times D : s(x, y) \leq t\}/\mu(D \times D)$$

where $\mu$ denotes the usual Lebesgue measure on $\mathbf{R}^{2d}$.

Then we have $\mu(D_\epsilon) = \mu\{(x, y) \in D \times D : s(x, y) \geq M - \epsilon\}$. For $d = 1$, $\mu(D_\epsilon)$ is the area of $D_\epsilon$ in Figure 6.6 .

It is clear now that the Gnedenko condition requires that the ratio of $\mu(D_{c\epsilon})$ to $\mu(D_\epsilon)$ tend to a constant $c^k$, as $\epsilon$ goes to 0. In other words, the behaviour of function $h(\epsilon) = \mu(D_\epsilon)$ must be like that of a power function as $\epsilon$ goes to zero.

Archetti, Betró and Steffé in [2] proposed a framework for global optimisation using random sampling of the objective function. They applied the theory of extreme value distributions to global maximisation problems, using the measure of the level set of the objective function to investigate the Gnedenko condition. They obtained a condition on the objective function which ensures that the measure of the level set possesses a certain limit property. This in turn ensures that the distribution of objective function values under random sampling satisfies the Gnedenko condition.

Our method is similar to the procedure of Archetti, Betró and Steffé, but here the function to be maximised is the absolute slope function $s$ of the original objective function $g$, not $g$ itself. Our aim has been to find a general condition on $g$ which ensures that the slope distribution satisfies the Gnedenko condition. The next lemma, expressed here in terms of the absolute slope function $s$, is due to Archetti, Betró and Steffé [2, §2 and Theorems 6 and 7]. It presents conditions on the absolute slope function $s$ which ensure that the cumulative distribution $F$ for the absolute value of slopes satisfies the Gnedenko condition. We are then able to prove our main theorem in the next subsection by showing that the conditions of that theorem ensure that those of the following lemma, Lemma 6.4.2, hold.

**Lemma 6.4.2** (1) *Suppose that* $s(x,y) = M + Q_{2p}((x,y) - (x_*, y_*)) + R(||(x,y) - (x_*, y_*)||^{2p})$, *where* $(x_*, y_*)$ *is an interior point of* $D \times D$ *at which* $s(x,y)$ *attains its unique global maximum* $M$, $Q_{2p}((x,y) - (x_*, y_*))$ *is a negative (for* $(x,y) \neq (x_*, y_*)$*) homogeneous polynomial of degree* $2p$, *for* $p$ *a natural number and* $R(\epsilon)/\epsilon \to 0$ *as* $\epsilon \to 0$. *Then* $\lim_{\epsilon \to 0+} \mu(D_\epsilon)/\epsilon^\alpha$ *exists and is finite and positive with* $\alpha = d/p$. *Here* $\mu$ *denotes Lebesgue measure on* $\boldsymbol{R}^{2d}$.

(2) *If* $\mu(D_\epsilon)$ *is such that*

$$\lim_{\epsilon \to 0+} \mu(D_\epsilon)/\epsilon^\alpha > 0$$

*for some* $\alpha > 0$, *then the Gnedenko condition holds for* $F(t) = \mu\{(x,y) \in D \times D : s(x,y) \leq t\}/\mu(D \times D)$, *with* $k = \alpha$.

The key to proving Lemma 6.4.2 (1) is the construction of level sets $D_\epsilon^1$ and $D_\epsilon^2$ such that $D_\epsilon^1 \subset D_\epsilon \subset D_\epsilon^2$. It is then shown that both $\lim_{\epsilon \to 0+} \mu(D_\epsilon^1)/\epsilon^\alpha$ and $\lim_{\epsilon \to 0+} \mu(D_\epsilon^2)/\epsilon^\alpha$ exist and are equal. The proof of the lemma is given in Appendix 2.

Remark: When $d = 1$, $D = [a,b]$, an interval. The requirement that $(x_*, y_*)$ be an interior point of $[a,b] \times [a,b]$ can be weakened to include the case that $(x_*, y_*) =$

$(a,a)$ or $(x_*,y_*)=(b,b)$ and the above Lemma remains true. This case is discussed in the proof of Lemma 6.4.2 in Appendix 2.

## 2. The one dimensional case

In this subsection we present our main results for the univariate case. We show that the Gnedenko condition holds for the absolute slope distribution $F$ of a wide class of univariate objective functions $g$. Partial results for the case of many variables will be given in the next subsection.

**Theorem 6.4.1** *Suppose that $g \in C^{2p+2}[a,b]$ for some natural number $p$, that there exists a unique $z \in (a,b)$ such that*

$$|g'(z)| = M = \max_{x\in(a,b)} |g'(x)| > 0,$$

*and that $g^{(i)}(z) = 0$ for $i = 2,3,\ldots,2p$ with $g^{(2p+1)}(z) \neq 0$. Given $\delta > 0$, let $(x,y)$ be a point chosen uniformly on $\{(x,y) \in [a,b]\times[a,b] : |x-y| \leq \delta\}$,*

$$s(x,y) = \begin{cases} |g(x)-g(y)|/|x-y| & \text{if } x \neq y \\ |g'(x)| & \text{if } x = y \end{cases}$$

*and $F$ denote the cdf of $s$. Then the Gnedenko condition holds for $F$ with $k = 1/p$. That is, for any $c > 0$,*

$$\lim_{\epsilon\to 0^+} \frac{1-F(M-c\epsilon)}{1-F(M-\epsilon)} = c^{1/p}.$$

Proof: Without loss of generality we can assume that $z = g(z) = 0$ and $g'(z) = M$. (This follows since we can let $h(x) = g(x-z)-g(z)$ and prove the result for $\pm h$ as needed.) We can then write $g(x)$ as

$$g(x) = Mx + a_{2p+1}x^{2p+1} + o(x^{2p+1})$$

by assumption, where $a_{2p+1} = g^{(2p+1)}(0)/(2p+1)!$ . Since $g'(0)$ is assumed to be the unique maximum of $g'$, we must have $a_{2p+1} < 0$. Also $\mathrm{diam}(D_\epsilon) \to 0$ as $\epsilon \to 0$ else the uniqueness assumption is contradicted.

Since $g'(0) = M > 0$, $g$ is increasing near 0, so we have

$$\frac{|g(x) - g(y)|}{|x - y|} = \frac{g(x) - g(y)}{x - y}$$

for any pair $x \neq y$ sufficiently close to 0. We need only prove that $(g(x)-g(y))/(x-y)$ is a function which satisfies the condition of Lemma 6.4.2 (1).

Since $\text{diam}(D_\epsilon) \to 0$ as $\epsilon \to 0$, for $\epsilon$ sufficiently small both $D_\epsilon$ and $D_{c\epsilon}$ are contained in $D$, Consider a point $(x, y)$ in $D_\epsilon$. Then the rearranged Taylor expansion of $g(x)$ about $y$ yields

$$\begin{aligned} & (g(x) - g(y))/(x - y) \\ = \; & g'(y) + \frac{g''(y)}{2!}(x - y) + \ldots + \frac{g^{(2p+1)}(y)}{(2p+1)!}(x - y)^{2p} + \frac{g^{(2p+2)}(\zeta_0)}{(2p+2)!}(x - y)^{2p+1} \end{aligned}$$

where $\zeta_0$ is between $x$ and $y$. Writing the Taylor expansion of $g(y), g'(y), \ldots, g^{(2p+1)}(y)$ about 0, we have

$$\begin{aligned} & (g(x) - g(y))/(x - y) \\ = \; & M + \frac{g^{(2p+1)}(0)}{(2p)!}y^{2p} + \frac{g^{(2p+2)}(\zeta_1)}{(2p+1)!}y^{2p+1} \\ & + \frac{1}{2!}[\frac{g^{(2p+1)}(0)}{(2p-1)!}y^{2p-1} + \frac{g^{(2p+2)}(\zeta_2)}{(2p+1)!}y^{2p}](x - y) + \ldots \\ & + \frac{1}{(2p+1)!}[\frac{g^{(2p+1)}(0)}{1!} + \frac{g^{(2p+2)}(\zeta_{2p+1})}{(2p+1)!}y](x - y)^{2p} + \frac{g^{(2p+2)}(\zeta_0)}{(2p+2)!}(x - y)^{2p+1} \\ = \; & M + \frac{g^{(2p+1)}(0)}{(2p+1)!}\sum_{k=0}^{2p}\frac{(2p+1)!}{(2p-k)!(k+1)!}y^{2p-k}(x - y)^k + R(\zeta_0, \zeta_1, \ldots, \zeta_{2p+1}, x, y) \end{aligned}$$

where $R(\zeta_0, \zeta_1, \ldots, \zeta_{2p+1}, x, y)$ equals

$$\frac{g^{(2p+2)}(\zeta_1)}{(2p+1)!}y^{2p+1} + \ldots + \frac{g^{(2p+2)}(\zeta_{2p+1})}{(2p+1)!}y(x - y)^{2p} + \frac{g^{(2p+2)}(\zeta_0)}{(2p+2)!}(x - y)^{2p+1}$$

and $\zeta_1, \ldots, \zeta_{2p+1}$ lie between 0 and $y$. Now

$$\begin{aligned} & \sum_{k=0}^{2p}\frac{(2p+1)!}{(2p-k)!(k+1)!}y^{2p-k}(x - y)^k \\ = \; & \sum_{k=0}^{2p} C_{k+1}^{2p+1} y^{2p-k}\sum_{i=0}^{k} C_i^k x^i(-1)^{k-i}y^{k-i} \\ = \; & \sum_{i=0}^{2p} x^i y^{2p-i}\sum_{k=i}^{2p}(-1)^{k-i}C_{k+1}^{2p+1}C_i^k \end{aligned}$$

where $C_i^k = \frac{k!}{i!(k-i)!}$ is the number of combinations of $i$ from $k$.

By using the generalised Vandermonde convolution formula of recurrence ([62, p.8]) it can be shown that $\sum_{k=i}^{2p}(-1)^{k-i}C_{k+1}^{2p+1}C_i^k = 1$, for $i = 0, 1, \ldots, 2p$.

In fact, the Vandermonde convolution formula is:

$$C_m^n = \sum_{k=0}^{m} C_{m-k}^{n-q}C_k^q.$$

Using the definition of combinations for a negative integer $-q$,

$$C_k^{-q} = (-1)^k C_k^{q+k-1},$$

the generalised Vandermonde convolution formula is

$$C_m^{n-q} = \sum_{k=0}^{m} C_{m-k}^n C_k^{-q} = \sum_{k=0}^{m}(-1)^k C_{m-k}^n C_k^{q+k-1} = \sum_{k=0}^{m}(-1)^k C_{n-m+k}^n C_k^{q+k-1}.$$

Now letting $n = 2p+1, m = 2p-i$ and $q = i+1$ in $C_m^{n-q} = \sum_{k=0}^{m}(-1)^k C_{n-m+k}^n C_k^{q+k-1}$ we have

$$\begin{aligned} 1 &= C_{2p-i}^{2p-i} \\ &= \sum_{k=0}^{2p-i}(-1)^k C_{k+i+1}^{2p+1} C_k^{k+i} \\ &= \sum_{k'=i}^{2p}(-1)^{k'-i} C_{k'+1}^{2p+1} C_{k'-i}^{k'} \\ &= \sum_{k=i}^{2p}(-1)^{k-i} C_{k+1}^{2p+1} C_i^k \end{aligned}$$

for $i = 0, 1, 2, \ldots, 2p$.

Therefore,

$$(g(x) - g(y))/(x-y) = M + \frac{g^{(2p+1)}(0)}{(2p+1)!}\sum_{i=1}^{2p} x^i y^{2p-i} + R(\zeta_0, \zeta_1, \ldots, \zeta_{2p+1}, x, y)$$

It is now readily checked that $s(x,y)$ is a function of the form described in Lemma 6.4.2 (1) with $d = 1$. Lemma 6.4.2 (1) thus shows that $\lim_{\epsilon \to 0+} \mu(D_\epsilon)/\epsilon^\alpha$ exists and is finite and positive with $\alpha = 1/p$. This limiting result rests on the

fact that we sample uniformly from a $\delta$-strip around the diagonal of $[a,b] \times [a,b]$, and that $s$ assumes its maximum value on the diagonal. Lemma 6.4.2 (2) then gives that the Gnedenko condition holds for $F(t) = \mu\{(x,y) \in D : s(x,y) \leq t\}/\mu(D)$ with $k = 1/p$. In a neighbourhood of $M$ this coincides with the cumulative distribution function of $s$, so Theorem 6.4.1 follows. □

The requirement that the first nonzero derivative at $z$ should occur at an odd order ensures that there is no contradiction to the assumption $|g'(z)| = M = \max_{x\in(a,b)} |g'(x)|$. We remark that the proof for an objective function $g$ containing only linear and pure cubic terms is much easier, as in this case the level sets $D_{c\epsilon}$ and $D_\epsilon$ are elliptical.

The following propositions, based on Theorem 6.4.1, show that the assumptions that $g'$ assumes its maximum absolute value at a unique point or an interior point can be removed.

**Proposition 6.4.2** *Suppose that $g \in C^{2p+2}(a,b)$ for some natural number $p$ with finitely many points $z_1, z_2, \ldots, z_k \in (a,b)$ such that $|g'(z_i)| = sup_{x\in(a,b)}|g'(x)| = M > 0$, and $g''(z_i) = \ldots = g^{2p}(z_i) = 0$ but $g^{(2p+1)}(z_i) \neq 0$ for $i = 1,2,\ldots,k$. Given $\delta > 0$, let $(x,y)$ be chosen uniformly on $\{(x,y) \in [a,b] \times [a,b] : |x-y| < \delta\}$, $s = |g(x) - g(y)|/|x-y|$ and $F$ denote the cumulative distribution function of slope $s$. Then the Gnedenko condition holds for $F$.*

Proof: For given $\epsilon > 0$ sufficiently small and $c > 0$, let $D_\epsilon$ and $D_{c\epsilon}$ be level sets of $s(x,y)$ on $[a,b] \times [a,b]$ as defined in the proof of Theorem 6.4.1. Since the $z_i$ are distinct there exist $\sigma_i > 0$ and disjoint intervals

$$I_i = (z_i - \sigma_i, z_i + \sigma_i) \quad \text{for} \quad i = 1,2,\ldots,k$$

such that $D_\epsilon \subset \cup_{i=1}^k I_i \times I_i$ and $D_{c\epsilon} \subset \cup_{i=1}^k I_i \times I_i$. Let

$$D_\epsilon^i = \{(x,y) \in I_i \times I_i : M - \epsilon < s(x,y) \leq M\} \quad \text{and}$$

$$D^i_{c\epsilon} = \{(x,y) \in I_i \times I_i : M - c\epsilon < s(x,y) \leq M\}$$

for $i = 1, 2, \ldots, k$. Then we have $D^i_\epsilon \subset I_i \times I_i$ and $D^i_{c\epsilon} \subset I_i \times I_i$. Therefore,

$$D_\epsilon = \cup_{i=1}^k D^i_\epsilon, \quad \text{with} \quad D^i_\epsilon \cap D^j_\epsilon = \emptyset \quad \text{for} \quad i \neq j, \quad \text{and}$$

$$D_{c\epsilon} = \cup_{i=1}^k D^i_{c\epsilon}, \quad \text{with} \quad D^i_{c\epsilon} \cap D^j_{c\epsilon} = \emptyset \quad \text{for} \quad i \neq j$$

For each $i$, $g$ is a function on $I_i$ satisfying the conditions of Theorem 6.4.1. Therefore, by the proof of Theorem 6.4.1, the corresponding slope function $s(x,y)$ on $I_i \times I_i$ satisfies the condition of Lemma 6.4.2 (1) with $d = 1$. Thus we have that $\lim_{\epsilon \to 0+} \mu(D^i_\epsilon)/\epsilon^{1/p}$ exists, with positive limit $h_i$. Therefore

$$\begin{aligned}
\lim_{\epsilon \to 0+} \frac{1 - F(M - c\epsilon)}{1 - F(M - \epsilon)} &= \lim_{\epsilon \to 0+} \frac{\mu(D_{c\epsilon})}{\mu(D_{c\epsilon})} \\
&= \lim_{\epsilon \to 0+} \frac{\sum_{i=1}^k \mu(D^i_{c\epsilon})}{\sum_{i=1}^k \mu(D^i_\epsilon)} = \frac{c^{1/p} \sum_{i=1}^k \lim_{\epsilon \to 0+} [\mu(D^i_{c\epsilon})/(c\epsilon)^{1/p}]}{\sum_{i=1}^k \lim_{\epsilon \to 0+} \mu(D^i_\epsilon)/\epsilon^{1/p}} \\
&= c^{1/p} \frac{\sum_{i=1}^k h_i}{\sum_{i=1}^k h_i} = c^{1/p}.
\end{aligned}$$

□

**Proposition 6.4.3** *Suppose that $g \in C^{2p+2}(a,b)$ for some natural number $p$ with $\lim_{\epsilon \to 0+} \frac{g(b) - g(b-\epsilon)}{\epsilon} = M > 0$, $g^{(2)}(b) = g^{(3)}(b) = \ldots = g^{(2p+1)}(b) = 0$, $g^{(2p+1)}(b) < 0$, where $g^{(k)}(b)$ denotes the left kth order derivative and $|g'(x)| < M$, on $(a,b)$. Given $\delta > 0$, let $(x,y)$ be chosen uniformly on $\{(x,y) \in [a,b] \times [a,b] : |x - y| < \delta\}$, $s = |g(x) - g(y)|/|x - y|$ and $F$ denote the cumulative distribution function of $s$. Then the Gnedenko condition holds for $F$.*

Proof: We may assume that $b = g(b) = 0$. If we take $\sigma > 0$ small enough we can have $g'$ increasing on $(-\sigma, 0)$. Define $G(x)$ to equal $g(x)$ on $[a,0]$, and $-g(-x)$ on $(0,\sigma]$. Then $G$ is a function on $[a, b+\sigma]$ which satisfies the conditions of Theorem

6.4.1. By the proof of Theorem 6.4.1, the slope function $s_G(x, y)$ of $G(x)$, determined on $[a, b+\sigma] \times [a, b+\sigma]$, has the form required by Lemma 6.4.2 (1).

Since $g(x) = G(x)$ for $x \in [a, b]$, the slope function $s_g(x, y)$ of $g$ satisfies $s_g(x, y) = s_G(x, y)$ for $(x, y) \in [a, b] \times [a, b]$. Noting the remark following Lemma 6.4.2, function $s_g$ satisfies the weakened condition of Lemma 6.4.2 (1). By the same argument as in Theorem 6.4.1, using Lemma 6.4.2 (2), the Gnedenko condition holds for $F$, the cumulative distribution function of the slope function $s_g$. □

It can be confirmed that all univariate test functions used in Section 6.3 satisfy the conditions of Theorem 6.4.1, Proposition 6.4.2 or 6.4.3.

## 3. The higher dimensional case

In this section, we investigate the validity of the Gnedenko condition for the absolute slope distribution when the objective function is of many variables.

The maximum absolute slope determined by a number of pairs of points drawn from the domain of the objective function will be used to estimate the Lipschitz constant, by the RW method described in Section 6.2. The Gnedenko limit involves the description of the level set (and its measure) for the slope function of the objective function near the maximum absolute slope itself. This is difficult unless we limit the class of objective functions. We start our discussion with linear (in fact affine) functions. A general differentiable objective function can be approximated by its tangent plane at the point where the maximum absolute slope occurs. Thus a study of linear objective functions might lead to results for general differentiable objective functions. Firstly, we present a theorem which shows that the Gnedenko condition holds for the slope of linear objective functions, under certain conditions. In the later part of this section, we will use the language of regularly varying functions and give a sufficient condition on an objective function to en-

sure that the cumulative distribution function $F$ of the slope function satisfies the Gnedenko condition.

Before we state the theorem for affine functions, we give the following definitions.

**Definition 6.4.1** *A compact convex subset $D$ of $\boldsymbol{R}^d$ is said to be* divisible *if $\partial D$, the boundary of $D$ can be written as two $(d-1)$ dimensional continuously differentiable functions on $\boldsymbol{R}^{d-1}$. That is, there exist continuously differentiable functions $f_i = f_i(x_1, \ldots, x_{d-1})$ for $i = 1, 2$ such that $\partial D = \partial D_1 \cup \partial D_2$ and the $d$th co-ordinate of $\partial D_i$ can be written as two continuously differentiable functions on $\boldsymbol{R}^{d-1}$:*

$$x_d = f_i(x_1, \ldots, x_{d-1}) \quad \textit{for} \quad i = 1, 2.$$

For example, for a $d$-dimensional sphere $D$ with radius $r$ : $\{(x_1, x_2, \ldots, x_d) : x_1^2 + x_2^2 + \ldots + x_{d-1}^2 + x_d^2 \leq r^2\}$, the boundary of the sphere $D$ is the graph of two $(d-1)$-dimensional functions

$$x_d = -(r^2 - x_1^2 - x_2^2 - \ldots - x_{d-1}^2)^{1/2} \quad \text{and}$$
$$x_d = (r^2 - x_1^2 - x_2^2 - \ldots - x_{d-1}^2)^{1/2}.$$

**Definition 6.4.2** *A cone in $\boldsymbol{R}^d$ has the form*

$$C = \{x = (x_1, \ldots, x_d) \in \boldsymbol{R}^d : \textstyle\sum_{i=1}^{d-1}(x_i - y_i)^2 \leq a(x_d - y_d)^2\}$$

*where $y = (y_1, \ldots, y_d)$ is the apex of $C$, and $a$ is a positive constant.*

The cross-section at $z = x_d$ orthogonal to the $d$th axis forms a $(d-1)$-dimensional sphere with radius $|x_d - y_d|$. The angle $\theta$ between the $d$th co-ordinate axis and any line from the apex to a point on the surface of the cone is called the half-apex angle of $C$. The "slope" or $\tan\theta$ equals $\sqrt{a}$. Figure 6.7 illustrates the case when $d = 3$.

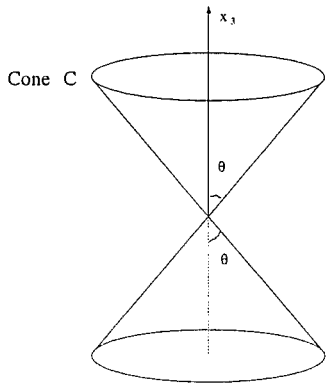



**Figure 6.7** A cone in $\mathbf{R}^3$ with half-apex angle $\theta$.

**Theorem 6.4.2** *Suppose $D \subset \boldsymbol{R}^d$ is a divisible compact convex subset, and*

$$g(x) = a_0 + a_1x_1 + a_2x_2 + \ldots + a_dx_d,$$

*where $a_i$ are constants, not all zero. Let $M = \max_{x\in D} || \bigtriangledown g(x)|| > 0$. For fixed $y \in \operatorname{int} D$, let $x$ be chosen uniformly on $D$ and let*

$$s(x,y) = \frac{|g(x) - g(y)|}{||x - y||}.$$

*Let $F$ denote the cumulative distribution function of $s$. Then the Gnedenko condition holds for $F$. That is, for any $c > 0$,*

$$\lim_{\epsilon\to 0+} \frac{1 - F(M - c\epsilon)}{1 - F(M - \epsilon)} = c^{(d-1)/2}.$$

Proof: Without loss of generality, we can assume that $a_i = 0$ for $i = 0, 1, \ldots, d-1$ and $a_d = M > 0$, that is, $g(x) = Mx_d$. This follows since we can align the $d$th axis $x_d$ with $(a_1, \ldots, a_d)$, the gradient of $g$, using an orthogonal transformation $(z_1, z_2, \ldots, z_d)' = Q(x_1, x_2, \ldots, x_d)'$ with orthogonal matrix $Q$ such that $z_d = \sum_{i=1}^d a_ix_i/(\sum_{i=1}^d a_i^2)^{1/2}$. Notice that this does not define Q uniquely if $d > 2$. We take an arbitrary one in this case. Under this transformation, $g(x) = a_0 + (\sum_{i=1}^d a_i^2)^{1/2}z_d$, so $g_1(z) = g(x) - a_0 = (\sum_{i=1}^d a_i^2)^{1/2}z_d$, has the form specified.

Given $\epsilon > 0$ denote

$$D_\epsilon(y) = \{x \in D : \frac{|g(x) - g(y)|}{||x - y||} \geq M - \epsilon\}.$$

We can also assume that $y = 0$ as the following proof will be same for any fixed $y \in$ int$D$.

Consider $x$ such that

$$\frac{|g(x) - g(y)|}{||x - y||} = \frac{|g(x)|}{||x||} \geq M - \epsilon.$$

Then we have

$$(Mx_d)^2 \geq (M - \epsilon)^2 \sum_{i=1}^{d} x_i^2, \quad \text{therefore}$$

$$\sum_{i=1}^{d-1} x_i^2 \leq \frac{M^2 - (M - \epsilon)^2}{(M - \epsilon)^2} x_d^2$$

So all $x \in D$ satisfying the above inequality form a cone $C$ in $\mathbf{R}^d$ with apex $y = 0$.

Let the half-apex angle be $\beta$ so $\tan \beta$ equals

$$a_\epsilon = \left(\frac{M^2 - (M - \epsilon)^2}{(M - \epsilon)^2}\right)^{1/2} = \frac{\epsilon^{1/2}(2M - \epsilon)^{1/2}}{M - \epsilon}.$$

Since $D$ is divisible, $\partial D = \partial D_1 \cup \partial D_2$ and there are two functions $f_1, f_2$, such that the $d$th co-ordinate of $\partial D_1, \partial D_2$, can be expressed as $x_d = f_1(x_1, \ldots, x_{d-1})$ and $x_d = f_2(x_1, \ldots, x_{d-1})$ respectively. Let

$$L_\epsilon = \{x \in D_\epsilon(0) : x_d \leq 0\} \quad \text{and} \quad U_\epsilon = \{x \in D_\epsilon(0) : x_d \geq 0\},$$

be the lower and upper part of $D_\epsilon(0)$. Then $L_\epsilon$ and $U_\epsilon$ are regions bounded by the cone $C$ and the graphs of $f_1$ and $f_2$. Figure 6.8 illustrates the situation when $d = 2$.

We have

$$\mu(D_\epsilon(0)) = \mu(L_\epsilon) + \mu(U_\epsilon)$$

where $\mu$ is Lebesgue measure on $\mathbf{R}^d$.

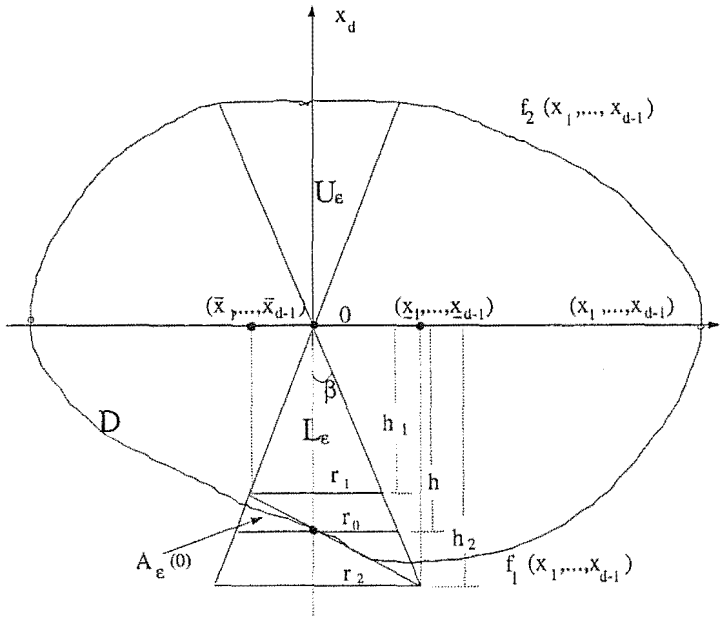



**Figure 6.8** Illustration of the level set $D_\epsilon(y)$, its lower and upper parts $L_\epsilon$ and $U_\epsilon$, the related heights $h_i$ and radii $r_i$ of the cone $C$, for the slope $s(x,y)$ of a linear function.

We deal now with the lower part $L_\epsilon$. The upper part $U_\epsilon$ can be discussed similarly. Let

$$A_\epsilon(0) = \{(x_1, \ldots, x_{d-1}, f_1(x_1, \ldots, x_{d-1})) : (x_1, \ldots, x_{d-1}, f_1(x_1, \ldots, x_{d-1})) \in L_\epsilon\},$$

the graph of the boundary function $f_1$ within the cone $C$ and denote by $\mathrm{Proj}A_\epsilon(0)$ the projection of $A_\epsilon(0)$ onto $\mathbf{R}^{d-1}$, the domain of $f_1$ in cone $C$. Let

$$\bar{f}_1 = \max_{x \in A_\epsilon(0)} f_1(x_1, \ldots, x_{d-1}) \quad \text{and} \quad \underline{f}_1 = \min_{x \in A_\epsilon(0)} f_1(x_1, \ldots, x_{d-1}).$$

We now construct two hypercones, one includes $L_\epsilon$ and the other is included in $L_\epsilon$.

Consider a hyperplane $H$ passing through $(0, \ldots, 0. f_1(0, \ldots, 0))$ with form

$$x_d - f_1(0, \ldots, 0) = b_1 x_1 + \ldots + b_{d-1} x_{d-1},$$

with normal $(b, -1) = (b_1, \ldots, b_{d-1}, -1)$ and $||b||$ bounded. From this class of hyperplanes we will choose one which suits our purpose.

The intersection of $H$ and $C$ forms a $(d-2)$-dimensional surface which has a highest point with $d$th coordinate $x_d = \bar{x}_d$ say, and lowest point with $d$th coordinate $x_d = \underline{x}_d$ say, respectively. Figure 6.9 illustrates the case when $d = 2$ and $y = (0, 0)$. Consider the intersection of the cone $C$ with the hyperplanes $x_d = \bar{x}_d$ and $x_d = \underline{x}_d$.

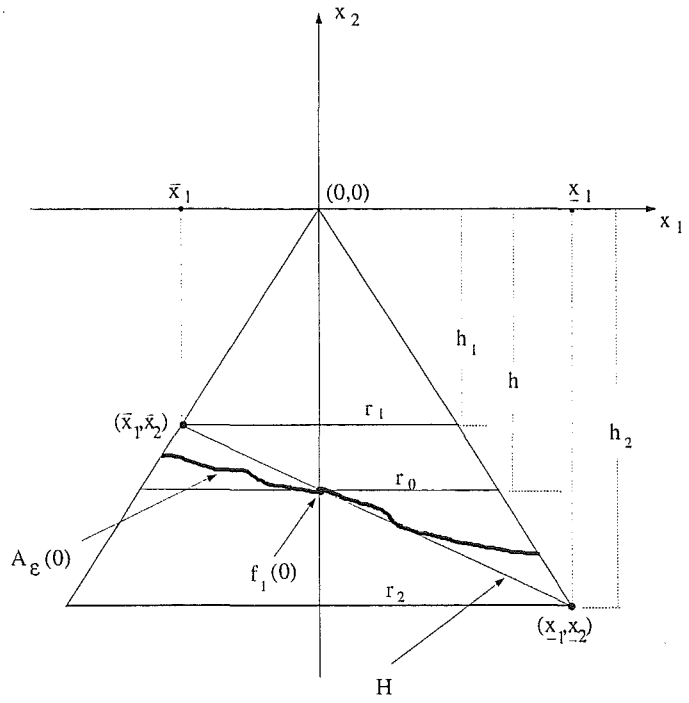



**Figure 6.9** Illustration of the construction of $H$. The related heights $h_i$ and radii $r_i$ of the cone $C$ for the slope $s(x, y)$ of a linear function are shown.

These cross sections of $C$ form two hyperspheres with radii $r_1$ and $r_2$ respectively. The cone $C$ and these two hyperspheres (as bases) in turn create two bounded cones $C_1(\epsilon)$ and $C_2(\epsilon)$ with heights $h_1$ and $h_2$. We also denote by $r_0$ the radius of the base of the cone $C_0(\epsilon)$ whose base passes through $(0, \ldots, 0, f_1(0, \ldots, 0))$ with height $h$(see Figure 6.9).

Since $\tan\beta$, the *slope* of the cone is $a_\epsilon$, we have

$$r_0 = a_\epsilon h, \quad r_1 = a_\epsilon h_1 \quad \text{and} \quad r_2 = a_\epsilon h_2$$

From the discussion above, $r_1$ and $r_2$, the base radii of $C_1(\epsilon)$ and $C_2(\epsilon)$, are

the minimum and maximum solutions of the following equations for $r$:

$$\begin{cases} \sum_{i=1}^{d-1} x_i^2 = a_\epsilon^2 x_d^2 \\ x_d - f_1(0,\ldots,0) = b_1 x_1 + \ldots + b_{d-1} x_{d-1} \\ r = \left(\sum_{i=1}^{d-1} x_i^2\right)^{1/2}. \end{cases}$$

The above equations have infinitely many solutions for $r = \left(\sum_{i=1}^{d-1} x_i^2\right)^{1/2}$ for $d > 2$; we seek the minimum and maximum such $r$. Notice that $b_1 x_1 + \ldots + b_{d-1} x_{d-1} = b \cdot (x_1, \ldots, x_{d-1})$, the dot product of $b = (b_1, \ldots, b_{d-1})$ and $(x_1, \ldots, x_{d-1})$, so we have

$$b_1 x_1 + \ldots + b_{d-1} x_{d-1} = ||b|| r \cos\phi$$

where $\phi$ is the angle between the vector $b$ and the vector $(x_1, \ldots, x_{d-1})$, with $0 \leq \phi \leq \pi$. We solve the above equations for maximum and minimum $r$ as follows: substituting $r = \left(\sum_{i=1}^{d-1} x_i^2\right)^{1/2}$ into the first equation, we have $r^2 = a_\epsilon^2 x_d^2$ which gives $x_d = -r/a_\epsilon$ (as $x_d \leq 0$). Putting $b_1 x_1 + \ldots + b_{d-1} x_{d-1} = ||b|| r \cos\phi$ into the second equation and replacing $x_d$ by $-r/a_\epsilon$, we have

$$-\frac{r}{a_\epsilon} = ||b|| r \cos\phi + f_1(0,\ldots,0)$$

or

$$r(1 + a_\epsilon ||b|| \cos\phi) = -a_\epsilon f_1(0,\ldots,0).$$

As $r$ and $a_\epsilon$ are positive and $a_\epsilon \longrightarrow 0$ as $\epsilon \to 0$, we have, for sufficiently small $\epsilon$,

$$r = \frac{a_\epsilon h}{1 + a_\epsilon ||b|| \cos\phi} = \frac{r_0}{1 + a_\epsilon ||b|| \cos\phi}$$

with the minimum and maximum such $r$ corresponding to $\phi = 0$ and $\phi = \pi$ respectively. That is,

$$r_1 = \frac{r_0}{1 + a_\epsilon ||b||} \quad \text{and} \quad r_2 = \frac{r_0}{1 - a_\epsilon ||b||}.$$

Then

$$h_1 = \frac{r_1}{a_\epsilon} = \frac{-f_1(0,\ldots,0)}{1 + a_\epsilon ||b||}, \quad h_2 = \frac{r_2}{a_\epsilon} = \frac{-f_1(0,\ldots,0)}{1 - a_\epsilon ||b||}.$$

As $f_1$ is continuously differentiable on $\mathrm{Proj}A_\epsilon(0)$, a compact subset of $\mathbf{R}^{d-1}$, it is Lipschitzian on $\mathrm{Proj}A_\epsilon(0)$. Thus there exists a $B > 0$, such that for $(\hat{x}_1, \ldots, \hat{x}_{d-1})$ and $(\check{x}_1, \ldots, \check{x}_{d-1})$, the maximiser and minimiser of $f_1$ over $\mathrm{Proj}A_\epsilon(0)$, we have

$$\bar{f}_1 - f_1(0, \ldots, 0) \leq B||(\hat{x}_1, \ldots, \hat{x}_{d-1})|| \quad \text{and} \quad f_1(0, \ldots, 0) - \underline{f}_1 \leq B||(\check{x}_1, \ldots, \check{x}_{d-1})||.$$

For $x \in \mathrm{Proj}A_\epsilon(0)$ and $\epsilon < \epsilon_0$, with $\epsilon_0$ a sufficiently small positive number, we choose $b$, such that $||b|| = B$, $b$ does not depend on $\epsilon$, and $a_\epsilon||b|| < 1$. Since $(\hat{x}_1, \ldots, \hat{x}_{d-1}), (\check{x}_1, \ldots, \check{x}_{d-1}) \in \mathrm{Proj}A_\epsilon(0)$, and $A_\epsilon(0)$ is inside the cone $C$, we have

$$\frac{||(\hat{x}_1, \ldots, \hat{x}_{d-1})||}{-\bar{f}_1} \leq a_\epsilon \quad \text{and} \quad \frac{||(\check{x}_1, \ldots, \check{x}_{d-1})||}{-\underline{f}_1} \leq a_\epsilon.$$

Therefore,

$$\bar{f}_1 - f_1(0, \ldots, 0) \leq B||(\hat{x}_1, \ldots, \hat{x}_{d-1})|| = ||b||\ ||(\hat{x}_1, \ldots, \hat{x}_{d-1})|| \leq -\bar{f}_1 a_\epsilon ||b|| \quad \text{and}$$

$$f_1(0, \ldots, 0) - \underline{f}_1 \leq B||(\check{x}_1, \ldots, \check{x}_{d-1})|| = ||b||\ ||(\check{x}_1, \ldots, \check{x}_{d-1})|| \leq -\underline{f}_1 a_\epsilon ||b||.$$

Hence

$$-h_1 = \frac{f_1(0, \ldots, 0)}{1 + a_\epsilon||b||} \geq \bar{f}_1 \quad \text{and} \quad -h_2 = \frac{f_1(0, \ldots, 0)}{1 - a_\epsilon||b||} \leq \underline{f}_1.$$

Thus

$$h_1 \leq -\bar{f}_1 \quad \text{and} \quad h_2 \geq -\underline{f}_1,$$

so the volume of $L_\epsilon$ is bounded by the volumes of $C_1(\epsilon)$ and $C_2(\epsilon)$ from above and below, that is

$$\mu(C_1(\epsilon)) \leq \mu(L_\epsilon) \leq \mu(C_2(\epsilon)) \quad \text{with}$$

$$\mu(C_1(\epsilon)) = \frac{\pi^{d/2} r_1^{d-1} h_1}{d\Gamma(d/2+1)}, \quad \text{where} \quad h_1 = -\overline{x}_d,$$

$$\mu(C_2(\epsilon)) = \frac{\pi^{d/2} r_2^{d-1} h_2}{d\Gamma(d/2+1)}, \quad \text{where} \quad h_2 = -\underline{x}_d.$$

For the above volume formula for $C_1(\epsilon)$ and $C_2(\epsilon)$, we refer the reader to [38]. We also have

$$\mu(C_0(\epsilon)) = \frac{\pi^{d/2} r_0^{d-1} h}{d\Gamma(d/2+1)} = \frac{\pi^{d/2} a_\epsilon^{d-1} h^d}{d\Gamma(d/2+1)}, \quad \text{where} \quad h = -f_1(0, \ldots, 0).$$

We now show that $\mu(C_i(\epsilon)) = \mu(C_0(\epsilon))(1 + O(\epsilon^{1/2}))$ for $i = 1, 2$. That is, the volumes of the hypercones $C_i(\epsilon)$ have the same lowest order as functions of $\epsilon$ as that of the hypercone $C_0$. Then we will be able to replace $\mu(C_i)$ by $\mu(C_0)$ in the required Gnedenko limit.

Recall that $a_\epsilon = \epsilon^{1/2}(2M - \epsilon)^{1/2}/(M - \epsilon)$, $r_1 = r_0/(1 + a_\epsilon||b||)$ and $r_2 = r_0/(1 - a_\epsilon||b||)$ We use the Taylor expansion for $(1 \pm a_\epsilon||b||)^{-1}$, a function of $\epsilon$, about $\epsilon$, to give

$$\begin{aligned}
& (1 \pm a_\epsilon||b||)^{-1} \\
= \; & 1 \mp a_\epsilon||b|| + (a_\epsilon||b||)^2 + \dots \\
= \; & 1 \mp \frac{\epsilon^{1/2}(2M - \epsilon)^{1/2}}{M - \epsilon}||b|| + \frac{\epsilon(2M - \epsilon)}{(M - \epsilon)^2}||b||^2 + \dots \\
= \; & 1 + O(\epsilon^{1/2}).
\end{aligned}$$

Therefore

$$r_i = r_0(1 + O(\epsilon^{1/2}))$$

and

$$h_i = \frac{r_i}{a_\epsilon} = \frac{r_0(1 + O(\epsilon^{1/2}))}{a_\epsilon} = h(1 + O(\epsilon^{1/2})) \quad \text{for} \;\; i = 1, 2,$$

which gives us

$$\begin{aligned}
\mu(C_i(\epsilon)) &= \frac{\pi^{d/2} r_i^{d-1} h_i}{d\Gamma(d/2 + 1)} \\
&= \frac{\pi^{d/2} r_0^{d-1} h}{d\Gamma(d/2 + 1)}(1 + O(\epsilon^{1/2}))^d \\
&= \frac{\pi^{d/2} r_0^{d-1} h}{d\Gamma(d/2 + 1)}(1 + O(\epsilon^{1/2})) \\
&= \mu(C_0(\epsilon))(1 + O(\epsilon^{1/2})) \quad \text{for} \;\; i = 1, 2.
\end{aligned}$$

Therefore $\mu(L_\epsilon) = \mu(C_0(\epsilon))(1 + O(\epsilon^{1/2}))$.

Similarly, replacing $\epsilon$ by $c\epsilon$ and defining terms in the same manner, we have

$$\mu(L_{c\epsilon}) = \mu(C_0(c\epsilon))(1 + O(\epsilon^{1/2}))$$

where

$$\mu(C_0(c\epsilon)) = \frac{\pi^{d/2} a_{c\epsilon}^{d-1} h^d}{d\Gamma(d/2+1)},$$

the Lebesgue measure of the cone and $a_{c\epsilon} = \dfrac{(c\epsilon)^{1/2}(2M - c\epsilon)^{1/2}}{M - c\epsilon}$.

Similar results can be obtained for $U_\epsilon$ and $U_{c\epsilon}$. That is, defining terms in the same manner, we arrive at

$$\mu(U_\epsilon) = \mu(CU_0(\epsilon))(1 + O(\epsilon^{1/2})) \quad \text{and} \quad \mu(U_{c\epsilon}) = \mu(CU_0(c\epsilon))(1 + O(\epsilon^{1/2})),$$

with

$$\mu(CU_0(\epsilon)) = \frac{\pi^{d/2} a_\epsilon^{d-1} h_u^d}{d\Gamma(d/2+1)} \quad \text{and} \quad \mu(CU_0(c\epsilon)) = \frac{\pi^{d/2} a_{c\epsilon}^{d-1} h_u^d}{d\Gamma(d/2+1)}.$$

Now

$$\begin{aligned}
& \lim_{\epsilon \to 0^+} \frac{1 - F(M - c\epsilon)}{1 - F_(M - \epsilon)} \\
= & \lim_{\epsilon \to 0^+} \frac{\mu(D_{c\epsilon}(0))}{\mu(D_\epsilon(0))} \\
= & \lim_{\epsilon \to 0^+} \frac{\mu(L_{c\epsilon}) + \mu(U_{c\epsilon})}{\mu(L_\epsilon) + \mu(U_\epsilon)} \\
= & \lim_{\epsilon \to 0^+} \frac{[\mu(C_0(c\epsilon)) + \mu(CU_0(c\epsilon))](1 + O(\epsilon^{1/2}))}{[\mu(C_0(\epsilon)) + \mu(CU_0(\epsilon))](1 + O(\epsilon^{1/2}))} \\
= & \lim_{\epsilon \to 0^+} \frac{\{\pi^{d/2} a_{c\epsilon}^{d-1}(h_l^d + h_u^d)\}/\{d\Gamma(d/2+1)\}}{\{\pi^{d/2} a_\epsilon^{d-1}(h_l^d + h_u^d)\}/\{d\Gamma(d/2+1)\}} \\
= & \lim_{\epsilon \to 0^+} \frac{(c\epsilon)^{(d-1)/2}(2M - c\epsilon)^{(d-1)/2}/(M - c\epsilon)^{(d-1)}}{(\epsilon)^{(d-1)/2}(2M - \epsilon)^{(d-1)/2}/(M - \epsilon)^{(d-1)}} \\
= & \; c^{(d-1)/2}
\end{aligned}$$

where $h_l$ and $h_u$ are the heights of the lower and upper cones respectively. □

Theorem 6.4.2 demonstrates that the Gnedenko condition holds for the cumulative distribution function of the slope (with one point fixed) of higher dimensional affine functions.

Computer tests were conducted for a number of linear functions using $y = 0$, $c = 2$ and 3, and various $d$ values. Figure 6.10 illustrates the results of a computer

trial to explore the Gnedenko limit for the test function $g(x_1, ...x_d) = x_1+x_2+...+x_d$ over the $d$-cube $D = \{x = (x_1, x_2, \ldots, x_d) : -1 \leq x_i \leq 1 \text{ for } 1 \leq i \leq d\}$ for $d = 2, 3, 5$ and $c = 2, 3$. For $c = 2$ and 3, we generate the absolute slope of function $g$ of 50000 pairs of points uniformly distributed on $D \times D$, with $\delta = 0.01$. For $\epsilon = 0.1, 0.099, 0.098, \ldots, 0.002, 0.001$, we use the numbers of absolute slopes whose value falls in the level set of $D_\epsilon$ and $D_{c\epsilon}$. The Gnedenko limit is then illustrated by plotting the (Gnedenko) ratio of these approximations against $\epsilon$.

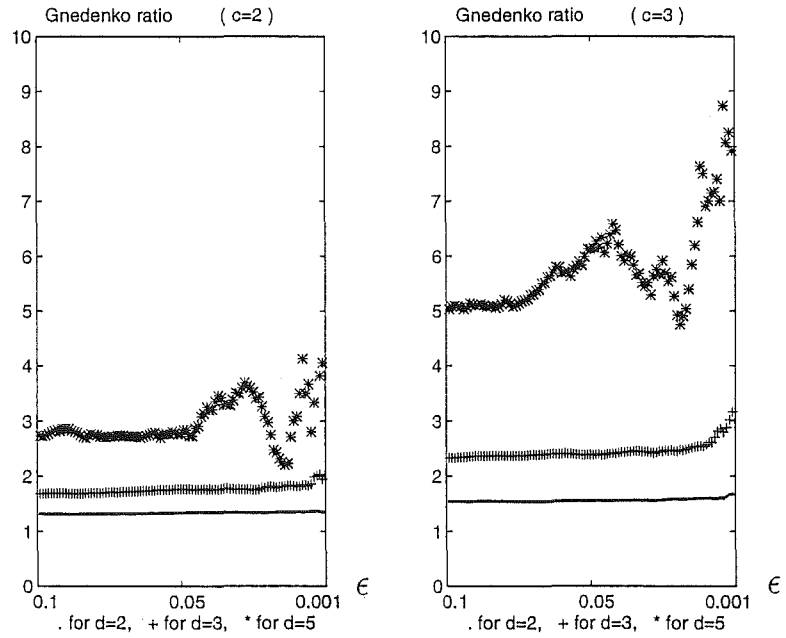



**Figure 6.10** Gnedenko limit for test function $g(x_1, ...x_d) = x_1 + x_2 + ... + x_d$ over the $d$-cube $\{x = (x_1, x_2, \ldots, x_d) : -1 \leq x_i \leq 1 \text{ for } 1 \leq i \leq d\}$, for $d = 2, 3, 5$ and $c = 2, 3$.

The above graphs show how the Gnedenko ratio changes as $\epsilon$ diminishes from 0.1 to 0.001 by a step 0.001. The results indicate that the power in the limit is $(d-1)/2$, and are consistent with the result of Theorem 6.4.2.

### Gnedenko condition and regularly varying functions

We now use the language of regularly varying functions to present a condition that ensures that the cumulative distribution function of the absolute slope function of

an objective function satisfies the Gnedenko condition. We give the definition of regularly varying functions first. The concepts and definitions of regularly varying functions can be found in [66] [74].

**Definition 6.4.3** *A univariate function $R$ is said to be a regularly varying function at 0 if it is real-valued, positive and measurable on $(0, A]$ with $A > 0$ and for each $x > 0$,*

$$\lim_{\epsilon \to 0^+} \frac{R(x\epsilon)}{R(\epsilon)} = x^k$$

*for some $k \in (-\infty, \infty)$, where $k$ is called the index of regular variation.*

Comparing the Gnedenko condition with the above definition of a regularly varying function, it is evident we can express the Gnedenko condition for a cumulative distribution function $F$ as: function $1 - F(M - x)$ is a regularly varying function at 0 with positive index.

In [18], de Haan discussed estimation of the minimum of a function using order statistics. A multidimensional regular variation function is used to give the condition on the objective function. For describing our result, we need Stam's definition of multivariate regular variation.

**Definition 6.4.4** *Suppose $C \subset \boldsymbol{R}^d$ is a cone, that is $\rho x \in C$ iff $x \in C$ for every $\rho > 0$. Suppose $\boldsymbol{1} \in C$, where $\boldsymbol{1} = (1, \ldots, 1)'$. Then a function $f : C \to (0, \infty)$ is said to be regularly varying on $C$ at $\boldsymbol{0}$ with limit function $\lambda(x) > 0$, if for all $x \in C$*

$$\lim_{\epsilon \to 0^+} \frac{f(\epsilon x)}{f(\epsilon \boldsymbol{1})} = \lambda(x),$$

Stam showed that the limit function $\lambda$ in this definition has the following property: for some $k \in \mathbf{R}$, (termed the index of function $\lambda(x)$),

$$\lambda(\rho x) = \rho^k \lambda(x), \quad \text{for all } x \in C \text{ and } \rho > 0.$$

de Haan in [17] showed that an equivalent way to express regular variation of a function is to say that there must exist a positive regularly varying function $U$ of dimension one with index $k$, such that

$$\lim_{\epsilon \to 0+} \frac{f(\epsilon x)}{U(\epsilon)} = \lambda(x) > 0, \quad \text{for all } \ x \in C.$$

The following theorem gives a sufficient condition on the absolute slope function of an objective function to ensure that $1 - F(M - x)$ is a regularly varying function at 0 with positive index $k$. This theorem and the proof are translations of a result in [18], for an objective function, to the absolute slope function of the objective function.

**Theorem 6.4.3** *Suppose $g$ is a differentiable function defined on $D \subset \boldsymbol{R}^d$,*

$$M = \max_{x \in D} || \bigtriangledown g(x)|| = || \bigtriangledown g(0)||, \quad \textit{and} \quad || \bigtriangledown g(x)|| < || \bigtriangledown g(0)|| \ \textit{for } x \neq 0.$$

*Given $\delta > 0$, let $(x_1, x_2)$ be a point chosen uniformly on $\{(x_1, x_2) \in D \times D : |x_1(i) - x_2(i)| \leq \delta, \ \textit{for } i = 1, \ldots, d\}$,*

$$s(x_1, x_2) = |g(x_1) - g(x_2)|/||x_1 - x_2|| \quad \textit{for} \quad x_1 \neq x_2$$

*and $F$ denote the cumulative distribution function of $s$. If $M - s(x_1, x_2)$ is a regularly varying function at $\boldsymbol{0}$ with positive index $k$, then $1 - F(M - t)$ is a regularly varying function of $t$ at 0 with positive index $2d/k$.*

Proof: Let $x = (x_1, x_2)$ and $f(x) = M - s(x)$. Since $f$ is a regularly varying function at $\mathbf{0}$ with positive index $k$, there exists a positive function $U$, such that

$$\lim_{\epsilon \to 0+} \frac{f(\epsilon x)}{U(\epsilon)} = \lambda(x) > 0, \quad \text{for all } \ x \in C \quad \text{and} \quad \lambda(\rho x) = \rho^k \lambda(x)$$

for some $k > 0$ and all $x \in C$, with $\rho > 0$. It follows that $\lim_{\rho \to 0+} \lambda(\rho x) = 0$, for any $x \in C$ and $x \neq 0$.

Let $B_y = \{u : \lambda(u) \leq y\}$. Then we have

$$B_{\epsilon y} = \epsilon^{1/k} B_y$$

for $\epsilon, y > 0$. Therefore $\mu(B_y) = c_0 y^{2d/k}$, where $c_0$ is a constant.

For $\epsilon, y > 0$,

$$\begin{aligned} 1 - F(M - yU(\epsilon)) &= \mu(\{u : s(u) \geq M - yU(\epsilon)\})/\mu(D \times D) \\ &= \mu(\{u : M - s(u) \leq yU(\epsilon)\})/\mu(D \times D) \\ &= \mu(\{u : f(u)/U(\epsilon) \leq y\})/\mu(D \times D) \\ &= \epsilon^{2d}\mu(\{u : f(\epsilon u)/U(\epsilon) \leq y\})/\mu(D \times D). \end{aligned}$$

Since

$$B_{(1-\delta)y} \subset \liminf_{\epsilon \to 0+} \{u : f(\epsilon u)/U(\epsilon) \leq y\}$$

$$\subset \limsup_{\epsilon \to 0+} \{u : f(\epsilon u)/U(\epsilon) \leq y\} \subset B_{(1+\delta)y}$$

for any $\delta > 0$, we have

$$\lim_{\epsilon \to 0+} \mu(\{u : f(\epsilon u)/U(\epsilon) \leq y\}) = \mu(B_y) = c_0 y^{2d/k}.$$

So

$$\lim_{\epsilon \to 0+} \{\epsilon^{-2d}[1 - F(M - yU(\epsilon))]\} = \frac{c_0 y^{2d/k}}{\mu(D \times D)}$$

for any $y > 0$. Let $y_1 = c\epsilon/U(\epsilon)$ and $y_2 = \epsilon/U(\epsilon)$ in $1 - F(M - yU(\epsilon))$ respectively.

Then we have

$$1 - F(M - c\epsilon) = \epsilon^{2d}\mu(\{u : f(\epsilon u)/U(\epsilon) \leq y_1\})/\mu(D \times D) \quad \text{and}$$

$$1 - F(M - \epsilon) = \epsilon^{2d}\mu(\{u : f(\epsilon u)/U(\epsilon) \leq y_2\})/\mu(D \times D).$$

Therefore

$$\begin{aligned} & \lim_{\epsilon \to 0+} \frac{1 - F(M - c\epsilon)}{1 - F(M - \epsilon)} \\ = & \lim_{\epsilon \to 0+} \frac{\epsilon^{2d}\mu(\{u : f(\epsilon u)/U(\epsilon) \leq y_1\})/\mu(D \times D)}{\epsilon^{2d}\mu(\{u : f(\epsilon u)/U(\epsilon) \leq y_2\})/\mu(D \times D)} \\ = & \lim_{\epsilon \to 0+} \frac{\epsilon^{2d} c_0 (c\epsilon/U(\epsilon))^{2d/k}/\mu(D \times D)}{\epsilon^{2d} c_0 (\epsilon/U(\epsilon))^{2d/k}/\mu(D \times D)} \\ = & \; c^{2d/k}. \end{aligned}$$

□

A sufficient condition for function $f(x_1, x_2) = M - s(x_1, x_2)$ to be a regularly varying function at $\mathbf{0}$ with positive index $k$ is that the limit

$$K(x_1, x_2) = \lim_{t \to 0^+} \frac{M - s(tx_1, tx_2)}{t}$$

exists and is positive for all $(x_1, x_2) \in \mathbf{R}^{2d} - \{\mathbf{0}\}$, as in this case the positive regularly varying function $U(t)$ of dimension one, as described in p.145, is $t$. Then $k = 1$. It is still an open problem to find conditions upon the objective function $g$ such that the difference between the maximum absolute slope $M$ and the absolute slope function $s(x_1, x_2)$ is a regularly varying function. Notice also that for the linear objective functions discussed in the previous subsection, Theorem 6.4.3 cannot apply as there are infinitely many points where the maximum absolute slope $M$ can occur.

## 6.5 Conclusion and comments

In this chapter, we presented a stochastic approach for estimating the Lipschitz constant of a function. It has been shown that for a wide class of univariate functions the slope distribution function satisfies the Gnedenko condition. Therefore they are suitable for applying the Reverse Weibull method of finding the Lipschitz constant. The method is clearly successful, but computationally intensive. For multidimensional functions a partial result is proved. Note that correct subset-specific Lipschitz constant estimates can lead to much improved numerical performance (see e.g [58, 60]).

Three directions for further research suggest themselves. Firstly, how does the idea used here fare for functions of more than one variable? We have obtained only limited results for this case. To find a class of functions for which the difference of the maximum absolute slope $M$ and the absolute slope function $s(x, y)$ is a mul-

tivariate regularly varying function seems not a trivial task. Secondly, how should the slope sample size $n$ and the maximum slope sample $m$ be chosen? Clearly the answer depends on the objective function $g$, its domain and the value of $\delta$. A further delicate investigation is needed to find the relationship between convergence speed and the sizes of $n$ and $m$ for a given function $g$. Thirdly, it remains to interlock the Lipschitz constant estimator with Lipschitz based optimisation algorithms. In particular it is planned to combine this idea with the Wood multidimensional bisection algorithm.

CHAPTER 7

# Summary

An investigation of Lipschitz based global optimisation problems formed the theme of this thesis. Following Hansen and Jaumard, Lipschitz global minimisation problems were placed into two major classes: finding the global minimum value $f_*$ of a Lipschitz continuous function $f$ and finding the localisation of the global minimisers of the function $f$. Several Lipschitz based algorithms for solving global optimisation problems have been investigated and a stochastic method for estimating the Lipschitz constant of a function was constructed.

In Chapter 2, a set of selected Lipschitz based algorithms was reviewed. For deterministic algorithms, the Piyavskii-Shubert algorithm, the axiomatic approach of Pintér and the Mladineo spherical generalisation of the Piyavskii-Shubert algorithm to higher dimensions were reviewed and discussed. The branch and bound framework of Horst and Tuy was also reviewed. For stochastic algorithms, pure random search and the pure adaptive search of Patel, Zabinsky and Smith were reviewed. These algorithms provided the bases of the investigations contained in this thesis.

In Chapter 3, some convergence properties of Lipschitz based algorithms were discussed. A lower bounding function of a Lipschitz continuous function $f$ was defined and the best lower bounding function of an objective function was presented.

A necessary condition on the objective function for finite convergence of an algorithm was obtained. These results are extensions of the one dimensional results of [31] to the higher dimensional case.

In Chapter 4, the Wood algorithm, multidimensional bisection, was discussed in detail together with a set of illustrations of the ideas in dimension two. Two major acceleration methods for multidimensional bisection were described. The performance of the multidimensional bisection algorithm with and without acceleration was investigated for a set of test functions. The Horst and Tuy branch and bound framework was extended to the language of covers. Multidimensional bisection was shown to fall into such a broadened branch and bound framework. Convergence of multidimensional bisection algorithm was discussed in two ways: range convergence and localisation convergence. A modification of the deepest point strategy for multidimensional bisection, using Basso's method, was defined and a localisation convergence result obtained which generalised Basso's result [8] to higher dimensions.

The aim of Chapter 5 was to provide a stochastic analogue of the Piyavskii-Shubert algorithm, one which can be readily implemented and also retain the linear complexity in dimension of pure adaptive search. A theoretical algorithm, somewhat adaptive search, was described. A special case of somewhat adaptive search, $\rho$-adaptive search, gave a spectrum of algorithms between the two extremes of pure random search and pure adaptive search. A readily implemented algorithm, pure localisation search, was defined and was shown to be a somewhat adaptive search algorithm for a limited class of functions. A comparison of pure random search, pure localisation search, pure adaptive search and the Piyavskii-Shubert algorithm was presented for a class of test functions which was "randomly" generated.

In Chapter 6, existing methods dealing with Lipschitz constant estimation

were reviewed. A stochastic method was constructed to estimate the Lipschitz constant of a function. The applicability of such a method was investigated. A class of univariate functions was shown to be such that the absolute slope distribution function satisfies the Gnedenko condition. This ensures the success of the estimation method. The higher dimensional case was also discussed and a partial result obtained for linear functions. Numerical results were provided which supported our theoretical results.

# APPENDIX 1

In this appendix, we give the proof of Theorem 5.5.1 developed by Baritompa in the team research process.

**Theorem 5.5.1** *Let $f$ be any function in $\mathcal{C}_h$, and for PLS with M=1, let $N$ be the number of iterations until the localisation becomes the level set. Then*

$$E[N] \; < \; 6 + 26/h$$

Proof: Take $f \in \mathcal{C}_h$. A typical situation which would arise when running PLS on $f$, once an evaluation is found less than $h$, is shown in Figure A.1.

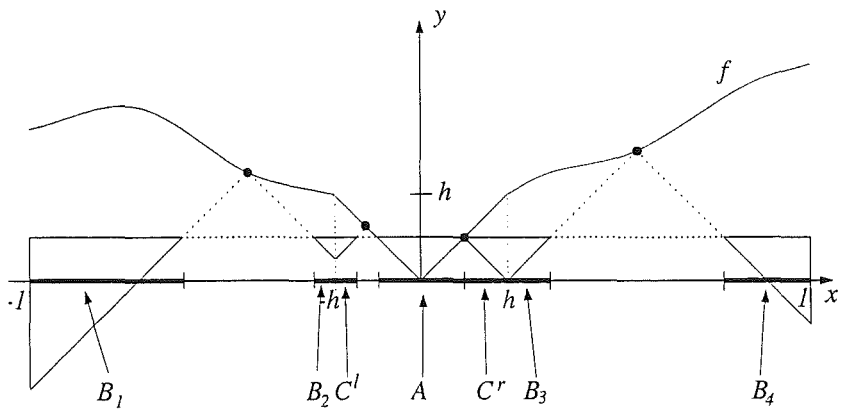



**Figure A.1** Running PLS on $f \in \mathcal{C}_h$: the four parts $A, B, C_l$ and $C_r$ of the localisation are shown. The situation illustrated is cap separated

The localisation, $L_k$, consists of four parts:

1. The level set, $A_k$.

2. A finite union of intervals, $B_k$, under the brim.

3. One interval, $C_k^l$, to the left of $A_k$ and under the cap.

4. One interval, $C_k^r$, to the right of $A_k$ and under the cap.

These sets are indicated in Figure A.1. Denote the total length of these sets by $a$,$b$,$c^l$, and $c^r$ respectively. We say at any stage that the run is *cap separated* if there has been an evaluation under the cap both to the left and to the right of the origin.

Four facts are needed in the final proof of the theorem. We present these now. The first three are readily shown; we give a proof for the fourth.

**Fact 1** *If the run at the kth iteration is cap separated, then the depth of the bracket over the complement of $A_k$ is less than or equal to $b + c^l + c^r$.*

**Fact 2** *Denote by $d$ the depth of the bracket over the complement of $A_k$. Then*

$$P[B_{k+1} \cup C_{k+1}^l \cup C_{k+1}^r = \phi \mid x_1, \ldots, x_k, \text{ and that } x_{k+1} \in A_k] \quad \geq \quad (a - 2d)/a$$

*where $a$ and $d$ are the values after the kth iteration.*

**Fact 3**

$$P[C_{k+1}^l = \phi \mid x_1, \ldots, x_k, \text{ and that } x_{k+1} \in C_k^l] \quad \geq \quad 1/2$$

**Fact 4** *Consider $t \in (0, h)$. If $\alpha_k$, the lowest evaluation immediately after the kth iteration, is less than $t$, then the number of further iterations under the brim is less than or equal to $2\left\lceil \frac{1-h}{h-t} \right\rceil$, where $\lceil x \rceil$ is the least integer greater than or equal to $x$.*

Proof of Fact 4: Suppose that after $k$ iterations we have $\alpha_k < t < h$. Let $z_1, z_2, \ldots$ be the later iterations of PLS which are in [h,1]. If $i < j$ then $z_j \notin \overline{B}_{h-t}(z_i)$, the closed interval of radius $h - t$ centered at $z_i$, so $\{\overline{B}_{(h-t)/2}(z_i) : i = 1, 2, \ldots\}$ is a mutually

disjoint collection of closed intervals whose union is a subset of $[h-(h-t)/2, 1+(h-t)/2]$. It follows that the collection must be finite, having say $m$ elements, and furthermore, that $m(h-t) < 1-t$. Thus $m$ is less than or equal to the biggest integer less than $(1-t)/(h-t) = (1-h)/(h-t)+1$. This is $\left\lceil \frac{1-h}{h-t} \right\rceil$. Fact 4 then follows by doubling this figure. □

The heart of the proof of the theorem rests in recognizing that if we count $N_t$, the number of iterations until the lowest known evaluation is less than $t$, and also the number of subsequent iterations, $N_p$, until we can be sure that the localisation is the level set, then $N_t + N_p$ is greater than or equal to $N$.

Following the iteration $N_t$ at which $\alpha_{N_t} < t$, we define five types of "progress" event which can occur. These are:

"$P_1$" Cap separation occurs for the first time at the $(k+1)^{th}$ iteration.

"$P_2$" $C_k^l \neq \phi$ and $C_{k+1}^l = \phi$

"$P_3$" $C_k^r \neq \phi$ and $C_{k+1}^r = \phi$

"$P_4$" $x_{k+1} \in B_k$

"$P_5$" $B_{k+1} \cup C_{k+1}^l \cup C_{k+1}^r = \phi$

Informally, a progress step is a movement towards the localisation becoming the level set, progress step five. Note that steps one, two and three can occur only once, while step four can occur at most $2\lceil(1-h)/(h-t)\rceil$ times. Thus, once there has been $2\lceil(1-h)/(h-t)\rceil + 3$ progress steps following iteration $N_t$, the localisation must equal the level set. If we let

$N_t$ = the number of iterations, $k$, until $\alpha_k < t$, and

$N_p$ = the number of iterations following the $(N_t)$th iteration to achieve $2\lceil(1-h)/(h-t)\rceil + 3$ progress steps,

we have $N \leq N_t + N_p$, so $E[N] \leq E[N_t] + E[N_p]$.

Certainly $E[N_t]$ is smaller for PLS than PRS on $f$. For PRS on $f$, the distribution of the number of iterations until a value less than or equal to $t$ is geometric, with probability $t$. Thus $E[N_t] \leq 1/t$.

We conclude the proof by showing that once we have $\alpha_{N_t} < t$, then the probability of a progress step is always at least $1/6$. The distribution of $N_p$ is negative binomial, so $E[N_p] \leq 6(2\lceil(1-h)/(h-t)\rceil + 3)$, whence

$$E[N] \leq \frac{1}{t} + 6(2\left\lceil\frac{1-h}{h-t}\right\rceil + 3) = 18 + \frac{1}{t} + 12\left\lceil\frac{1-h}{h-t}\right\rceil$$

Putting $t = h/2$ demonstrates the statement in the theorem.

In order to show that the probability of progress is always greater than or equal to $1/6$, we consider three cases. We suppose we have an initial segment of $x_1, \ldots, x_k$, and $\alpha_k \leq t$.

Case 1: The bracket is not cap separated. Then

$$\begin{aligned}
&P[\text{progress at } (k+1)\text{st iteration}] \\
&\geq P[P_1] + P[x_{k+1} \in C_k^l \text{ and } P_2] + P[x_{k+1} \in C_k^r \text{ and } P_3] + P[P_4] \\
&\geq \frac{a/2}{a + c^l + c^r + b} + \frac{c^l/2}{a + c^l + c^r + b} + \frac{c^r/2}{a + c^l + c^r + b} + \frac{b}{a + c^l + c^r + b} \\
&\geq \frac{1}{2}
\end{aligned}$$

Case 2: The bracket is cap separated, and $\dfrac{a}{a + c^l + c^r + b} \geq \dfrac{2}{3}$. Then

$$\begin{aligned}
&P[\text{progress at } (k+1)\text{st iteration}] \\
&\geq P[x_{k+1} \in A_k \text{ and } P_5] + P[x_{k+1} \in C_k^l \text{ and } P_2] + P[x_{k+1} \in C_k^r \text{ and } P_3] + P[P_4] \\
&\geq \frac{a - 2(c^l + c^r + b)}{a + c^l + c^r + b} + \frac{c^l/2}{a + c^l + c^r + b} + \frac{c^r/2}{a + c^l + c^r + b} + \frac{b}{a + c^l + c^r + b} \\
&= \frac{a - 3/2(c^l + c^r + b) + b/2}{a + c^l + c^r + b} \\
&\geq \frac{a}{a + c^l + c^r + b} - \frac{3}{2}\frac{c^l + c^r + b}{a + c^l + c^r + b} \\
&\geq \frac{2}{3} - \frac{3}{2}\cdot\frac{1}{3} = \frac{1}{6}
\end{aligned}$$

Case 3: The bracket is cap separated, and $\frac{a}{a+c^l+c^r+b} < \frac{2}{3}$. Then

$$
\begin{aligned}
& P[\text{progress at } (k+1)\text{st iteration}] \\
\geq\ & P[x_{k+1} \in C_k^l \text{ and } P_2] + P[x_{k+1} \in C_k^r \text{ and } P_3] + P[P_4] \\
\geq\ & \frac{c^l/2}{a+c^l+c^r+b} + \frac{c^r/2}{a+c^l+c^r+b} + \frac{b}{a+c^l+c^r+b} \\
\geq\ & \frac{1}{2} \cdot \frac{c^l+c^r+b}{a+c^l+c^r+b} \\
\geq\ & \frac{1}{2} \cdot \frac{1}{3} = \frac{1}{6}
\end{aligned}
$$

□

# APPENDIX 2

In this appendix, we give the proof of Lemma 6.4.2. To prove Lemma 6.4.2, we need the following lemma which is due to Archetti, Betró and Steffé. (see [2]).

**Lemma A2.1** *Let $Q_{2p}(x)$ for $x \in \boldsymbol{R}^d$ be a homogeneous negative polynomial (for $x \neq 0$) of degree $2p$. Then $\mu\{x \in \boldsymbol{R}^d : Q_{2p}(x) > -\epsilon\} = \epsilon^{d/(2p)}v$, for $\epsilon > 0$ with $v = \mu\{x \in \boldsymbol{R}^d : Q_{2p}(x) > -1\}$.*

Proof: Note that $\mu\{x \in \mathbf{R}^d : Q_{2p}(x) > -\epsilon\} = \mu\{x \in \mathbf{R}^d : (1/\epsilon)Q_{2p}(x) > -1\}$

$$= \mu\{x \in \mathbf{R}^d : Q_{2p}(\frac{x}{\epsilon^{1/(2p)}}) > -1\}$$

$$= \mu\{(1/\epsilon^{1/(2p)})u \in \mathbf{R}^d : Q_{2p}(u) > -1\}$$

$$= (1/\epsilon^{d/(2p)})\mu\{u : Q_{2p}(u) > -1\}.$$ □

Remark: If we change the statement from $x \in \mathbf{R}^d$ to $x \in C$, where $C$ is a cone with apex at 0 (that is, $x \in C$ iff $tx \in C$ for every $t > 0$) then this lemma remains true.

For completeness, we restate Lemma 6.4.2.

**Lemma 6.4.2** (1) *Suppose that $s(x,y) = M + Q_{2p}((x,y) - (x_*, y_*)) + R(||(x,y) - (x_*, y_*)||^{2p})$, where $(x_*, y_*)$ is an interior point of $D \times D$ at which $s(x,y)$ attains its unique global maximum $M$, $Q_{2p}((x,y) - (x_*, y_*))$ is a negative (for $(x,y) \neq (x_*, y_*)$) homogeneous polynomial of degree $2p$, for $p$ a natural number and $R(\epsilon)/\epsilon \to 0$ as $\epsilon \to 0$. Then $\lim_{\epsilon \to 0^+} \mu(D_\epsilon)/\epsilon^\alpha$ exists and is finite and positive with $\alpha = d/p$. Here $\mu$ denotes Lebesgue measure on $\boldsymbol{R}^{2d}$.*

(2) *If $\mu(D_\epsilon)$ is such that*

$$\lim_{\epsilon \to 0^+} \mu(D_\epsilon)/\epsilon^\alpha > 0$$

*for some* $\alpha > 0$, *then the Gnedenko condition holds for* $F(t) = \mu\{(x,y) \in D \times D : s(x,y) \leq t\}/\mu(D \times D)$.

The following proof is given in [2].

Proof: (1) Let $z = (x,y)$ be a $2d$-dimensional vector and $z_* = (x_*, y_*)$, so $s(x,y) = s(z)$. We have

$$s(z) = M + Q_{2p}(z - z_*) + o(||z - z_*||^{2p}),$$

and $Q_{2p}(z - z_*)$ a negative homogeneous polynomial, for any fixed $t > 0$. We can choose $\delta > 0$, such that for $||z - z_*|| < \delta$ we have $o(||z - z_*||) < t$. When $t$ is small enough, let

$$Q_t^+(z - z_*) = Q_{2p}(z - z_*) + t||z - z_*||^{2p} \quad \text{and}$$

$$Q_t^-(z - z_*) = Q_{2p}(z - z_*) - t||z - z_*||^{2p},$$

where $Q_t^+$ and $Q_t^-$ are both negative homogeneous functions of degree $2p$, in the vicinity of $z_*$. For $||z - z_*|| < \delta$, we have

$$Q_t^-(z - z_*) \leq s(z) - M \leq Q_t^+(z - z_*).$$

Recall that

$$\mu(D_\epsilon) = \mu\{z \in R^{2d} : s(z) \geq M - \epsilon\}$$

and define

$$h_t^+(\epsilon) = \mu\{z \in R^{2d} : Q_t^+(z - z_*) > -\epsilon\},$$

$$h_t^-(\epsilon) = \mu\{z \in R^{2d} : Q_t^-(z - z_*) > -\epsilon\}.$$

The uniqueness of $z_*$ and the negativeness of $Q_t^+$ and $Q_t^-$ give, for a sufficiently small $\epsilon$, say $\epsilon \leq \epsilon_\delta$ the following relationships:

$$\{z \in \mathbf{R}^{2d}: Q_t^-(z - z_*) > -\epsilon\} \subset \{z \in \mathbf{R}^{2d}: s(z) - M > -\epsilon\}$$
$$\subset \{z \in \mathbf{R}^{2d}: Q_t^+(z - z_*) > -\epsilon\}$$

$$\subset \{z \in \mathbf{R}^{2d}: ||z - z_*|| < \delta\}.$$

Hence, for every $\epsilon \in [0, \epsilon_\delta]$, we have

$$h_t^-(\epsilon) \leq \mu(D_\epsilon) \leq h_t^+(\epsilon).$$

The above lemma, Lemma A2.1 ensures that

$$h_t^+(\epsilon) = \epsilon^{d/p} h_t^+(1) = \epsilon^{d/p} v_t^+,$$

$$h_t^-(t) = \epsilon^{d/p} h_t^-(1) = \epsilon^{d/p} v_t^-.$$

Therefore, $v_t^- \leq \frac{\mu(D_\epsilon)}{\epsilon^{d/p}} \leq v_t^+$, and

$$v_t^- \leq \lim_{\epsilon \to 0^+} \inf \frac{\mu(D_\epsilon)}{\epsilon^{d/p}} \leq \lim_{\epsilon \to 0^+} \sup \frac{\mu(D_\epsilon)}{\epsilon^{d/p}} \leq v_t^+.$$

Let $\{t_k\}$ be a monotonically decreasing sequence tending to zero, and let

$$E_k = \{z : Q_{t_k}^+(z - z_*) > -1\}.$$

It follows that

$$E_k \supset E_{k+1} \supset E, \quad E = \{z : Q_{2p}(z - z_*) > -1\},$$

hence

$$\cap_{k=1}^{\infty} E_k \supset E.$$

In fact $\cap_{k=1}^{\infty} E_k = E$. To prove this, if there exists a point $\bar{z} \in \cap_{k=1}^{\infty} E_k$, $\bar{z} \notin E$ then

$$Q_{2p}(z - z_*) > -1 - t||z - z_*||^{2p},$$

for every $t$, and $Q_{2p}(z - z_*) < -1$, which is a contradiction. Therefore,

$$v_t^+ = \mu(E_k) \longrightarrow \mu(E) = v.$$

with $v = \mu\{z : Q_{2p}(z - z_*) > -1\}$. Similarly we can have $v_\epsilon^- \longrightarrow v$. Then

$$0 < \lim_{\epsilon \to 0^+} \inf \frac{\mu(D_\epsilon)}{\epsilon^{d/p}} = \lim_{\epsilon \to 0^+} \sup \frac{\mu(D_\epsilon)}{\epsilon^{d/p}} < \infty.$$

Therefore $\lim_{\epsilon \to 0^+} \mu(D_\epsilon)/\epsilon^{d/p}$ exists and is positive.

(2) If $\mu(D_\epsilon)$ satisfies

$$\lim_{\epsilon \to 0^+} \mu(D_\epsilon)/\epsilon^{\alpha} = H > 0 \quad \text{with positive } \alpha,$$

then

$$\begin{aligned}
& \lim_{\epsilon \to 0^+} \frac{1 - F(M - c\epsilon)}{1 - F(M - \epsilon)} \\
= & \lim_{\epsilon \to 0^+} \frac{\mu(D \times D) - \mu\{(x,y) \in D \times D : s(x,y) \leq M - c\epsilon\}}{\mu(D \times D) - \mu\{(x,y) \in D \times D : s(x,y) \leq M - \epsilon\}} \\
= & \lim_{\epsilon \to 0^+} \frac{\mu\{(x,y) \in D \times D : s(x,y) > M - c\epsilon\}}{\mu\{(x,y) \in D \times D : s(x,y) > M - \epsilon\}} \\
= & \lim_{\epsilon \to 0^+} \frac{\mu(D_{c\epsilon})}{\mu(D_\epsilon)} \\
= & \ c^{\alpha}.
\end{aligned}$$

□

Remark: When $d = 1$ and $D = [a, b]$, if $(x_*, y_*)$ is one of the vertices $(a, a)$ or $(b, b)$, then $s(x, y)$ is defined on a cone $C$ so Lemma A2.1 remains true, and the proof of Lemma 6.4.2 is unaffected. Therefore, the requirement that $(x_*, y_*)$ be an interior point of $[a, b] \times [a, b]$ can be weakened to include the case where $(x_*, y_*) = (a, a)$ or $(x_*, y_*) = (b, b)$.

# Figures

# Tables

# References


[1] Anderssen, R.S. (1972), "Global optimization", in *Optimization*, 26-48, edited by Anderssen, R.S., Jennings, L.S. and Ryan, D.M., University of Queensland Press, Queensland.

[2] Archetti, F., Betró, B. and Steffé, S. (1976), "A theoretical framework for global optimization via random sampling", in *Studies on Stochastic Optimization*, (Quaderno dei gruppi di recerca mathematica del C.N.R.), University of Pisa.

[3] Archetti, F. and Betró, B. (1978), "A priori analysis of deterministic strategies for global optimization problems", In *Towards Global Optimization 2*, L.C.W. Dixon and G.P. Szegö, North Holland, Amsterdam.

[4] Baba, N. (1981), "Convergence of a random optimization method for constrained optimization problems", *Journal of Optimization Theory and Applications* **33**, 451-461.

[5] Baritompa, W. P. (1994), "Multidimensional bisection: a dual viewpoint", *Computers and Mathematics with Applications* **27**, 11-22.

[6] Baritompa, W.P. (1993), "Customizing methods for global optimization - a geometric viewpoint", *Journal of Global Optimization* **3**, 193-212.

[7] Baritompa, W.P., Zhang, B.P. Mladineo, R.H., Wood, G R. and Zabinsky, Z.B. (1995), "Towards Pure Adaptive Search: A general framework and a one-dimensional realization", *Journal of Global Optimization* **7**, 93-110.

[8] Basso, P. (1982), "Iterative methods for the localization of the global maximum", *SIAM Journal on Numerical Analysis* **19**, 781-792.

[9] Bogomolov, N.A., and Karmanov, V.G. (1977), "Convergence of a random search method for finding stationary points of the general nonlinear programming problem". Bulletin of the Moscow State University, Ser. 15, *Computational Mathematics and Cybernetics* **1**, 20-26.

[10] Boon, J.G., Pintér, J. and Somlyódy, L. (1989), "A new stochastic approach for controlling point source river pollution", *Publications of the International Association of Hydrologic Science* **180**, 241-249.

[11] Brooks, S.H. (1958), " Discussion of random methods for locating surface maxima", *Operations Research* **6**, 244-251.

[12] Chuyan, O.R. and Sukharev, A.G. (1990), "On adaptive and nonadaptive stochastic and deterministic algorithms", *Journal of Complexity* **6**, 119-127.

[13] Danilin, Y.M. (1971), "Estimation of the efficiency of an absolute-minimum-finding algorithm", *U.S.S.R. Computational Mathematics and Mathematical Physics* **11**, 261-267.

[14] Denisov, D.V. (1978), "Random search for constrained minimization", *USSR Computational Mathematics and Mathematical Physics* **18**, 1103-1111.

[15] Devroye, L. (1978), "Progressive global random search of continuous functions", *Mathematical Programming* **15**, 330-342.

[16] Dixon, I.D.W. and Szegö, G.P. (1975), *Towards global optimisation*, North-Holland Publishing Company, Amsterdam.

[17] de Haan, L. and Resnick, S.I. (1987), "On regular variation of probability densities", *Stochastic Processes and their Applications* **25**, 83-93.

[18] de Haan, L. (1981), "Estimation of the minimum of a function using order statistics", *Journal of the American Statistical Association* **76**, 467-469.

[19] Evtushenko, Y.G. (1971), "Numerical methods for finding global extrema (case of non-uniform mesh)", *U.S.S.R. Computational Mathematics and Mathematical Physics* **11**, 38-54.

[20] Floudas, C.A. and Pardalos, P.M. (1992), *Recent advances in global optimization*, Princeton Series in Computer Science, Princeton University Press.

[21] Galambos, J. (1987), *The asymptotic theory of extreme order statistics*, second edition, Krieger, Florida.

[22] Galperin, E.A. (1985), "The cubic algorithm", *Journal of Mathematical Analysis and Applications* **112**, 635-640.

[23] Galperin, E.A. (1987), "The beta-algorithm", *Journal of Mathematical Analysis and Applications* **126**, 455-468.

[24] Galperin, E.A. and Zheng, Q. (1987), "Nonlinear observation via global optimization methods: Measure theory approach", *Journal of Mathematical Analysis and Applications* **54**, 63-92.

[25] Gnedenko, V.B. (1943), "Sur la distribution limite du terme maximum d'une série aléatoire", *Annals of Mathematics* **44**, 423-453.

[26] Gourdin, E., Hansen, P. and Jaumard, B. (1994), "Finding maximum likelihood estimators for the three-parameter Weibull distribution", *Journal of Global Optimization* **5**, 373-397.

[27] Graf, S., Mauldin, D.R. and Williams, S.C. (1986), "Random homeomorphisms", *Advances in Mathematics* **60**, 239-359.

[28] Hansen, P. and Jaumard, B. (1994), "Lipschitz optimization", *Technical Report G-94-22*, GERAD, École Polytechnique Université McGill.

[29] Hansen, P., Jaumard, B. and Lu, S.H. (1991), "On the number of iterations of Piyavskii's global optimization algorithm", *Mathematics of Operations Research* **16**, 334-350.

[30] Hansen, P., Jaumard, B. and Lu, S.H. (1992), "On the use of estimates of the Lipschitz constant in global optimization", *Journal of Optimization Theory and Applications* **75**, 195-200.

[31] Hansen, P., Jaumard, B. and Lu, S.H. (1992), "Global optimization of univariate Lipschitz functions: I. Survey and properties", *Mathematical Programming* **55**, 251-272.

[32] Hansen, P., Jaumard, B. and Lu, S.H. (1992), "Global optimization of univariate Lipschitz functions: II. New algorithms and computational comparison", *Mathematical Programming* **55**, 273-292.

[33] Horst, R. and Pardalos, P.M. (1994), *Handbook of Global Optimization*, Kluwer, Dordrecht.

[34] Hendrix, E.M.T. and Pintér, J. (1991), " An application of Lipschitzian global optimization to product design", *Journal of Global Optimization* **1**, 389-401.

[35] Horst, R. and Tuy, H. (1990), *Global optimization: deterministic approaches*, Springer-Verlag, Berlin.

[36] Horst, R. and Tuy H. (1987), "On the convergence of global methods in multiextremal optimization", *Journal of Optimization Theory and Applications* **54**, 253-271.

[37] Jaumard, B., Herremann, T. and Ribault, H. (1994), "An on-line cone intersection algorithm for global optimization of multivariate Lipschitz functions", working paper.

[38] Kendall, M.G. (1961), *A course in the geometry of n dimensions*, Charles Griffin & Company Limited, London.

[39] Maradas, C.D. and Floudas, C.A. (1994), "Global minimum potential energy conformations of small molecules", *Journal of Global Optimization* **4**, 135-170.

[40] Marti, K. (1980), "Random search in optimization problems as a stochastic decision process (adaptive random search)", *Methods of Operations Research* **36**, 223-234.

[41] Mayne, D.Q. and Polark, E. (1984), "Outer approximation algorithm for non-differentiable optimization problems", *Journal of Optimization Theory and Applications* **42**, 19-30.

[42] Meewella, C.C. and Mayne, D.Q. (1988), "An algorithm for global optimization of Lipschitz functions", *Journal of Optimization Theory and Applications* **57**, 307-323.

[43] Mladineo, R.H. (1986), "An algorithm for finding the global maximum of a multimodal, multivariate function", *Mathematical Programming* **34**, 188-200.

[44] Mladineo, R.H. (1992), "Convergence rates of a global optimization algorithm", *Mathematical Programming* **54**, 223-232.

[45] Mladineo, R.H. (1992), "Stochastic minimization of Lipschitz functions", in *Recent Advances in Global Optimization*, 369-383, Princeton Series in Computer Science, Princeton University Press.

[46] Mladineo, R.H. (1992), "Randomized cone algorithm", Working paper.

[47] Moore, R. E. (1966), *Interval analysis*, Prentice-Hall, Englewood Cliffs, N.J.

[48] Panchang, V.G. and Gupta, R.C. (1989), "On the determination of three-parameter Weibull MLE's", *Communications in Statistics B : Simulation and Computation* **18**, 1037-1057.

[49] Patel, N., Smith, R.L. and Zabinsky, Z.B. (1988), "Pure adaptive search in Monte Carlo optimization", *Mathematical Programming* **43**, 317-328.

[50] Patel, N. and Smith, R.L. (1983), "The asymptotic extreme value distribution of the sample minimum of a concave function under linear constraints", *Operations Research* **31**, 789-794.

[51] Pintér, J. (1982), "Hybrid procedures for solving non-smooth, stochastic constrained optimization problems", Bulletin of the Moscow State University, Ser. 15, *Computational Mathematics and Cybernetics* **1**, 39-49.

[52] Pintér, J. (1983), "A unified approach to globally convergent one-dimension optimization algorithms', *Technical Report IAMI-83.5*, Istituto per le applicazioni della Matematica e dell'Informatica, CNR, Milano.

[53] Pintér, J. (1984), "Convergence properties of stochastic optimization procedures", *Optimization* **15**, 405-427.

[54] Pintér, J. (1986), "Global optimization on convex sets", *Operations Research Spektrum* **6**, 197-202.

[55] Pintér, J. (1986), "Extended univariate algorithms for $n$-dimensional global optimization", *Computing* **36**, 91-103.

[56] Pintér, J. (1986), "Globally convergent methods for n-dimensional multi-extremal optimization", *Optimization* **17**, 187-202.

[57] Pintér, J. Szabó, J. and Somlyódy, L. (1986), "Multiextremal optimization for calibrating water resources models", *Environmental Software* **1**, 98-105.

[58] Pintér, J. (1988), "Branch-and-bound methods for solving global optimization problems with Lipschitzian structure", *Optimization* **19**, 101-110.

[59] Pintér, J. (1990), "Model calibration: Problem statement, solution method and implementation", Research Report 90.024, Rijkswaterstaat RIZA, Lelystad.

[60] Pintér, J. (1992), "Convergence qualification of partition algorithms in global optimization", *Mathematical Programming* **56**, 343-360.

[61] Piyavskii, S.A. (1972), "An algorithm for finding the absolute extremum of a function", *U.S.S.R Computational Mathematics and Mathematical Physics* **12**, 57-67.

[62] Riordan, J. (1968), *Combinatorial Identities*, Wiley, New York.

[63] Rinnooy Kan, A.H.G. and Timmer G.T. (1986), "Global optimization", *Technical Report 8612/A*, Econometric Institute, Erasmus University, Rotterdam.

[64] Romeijn, H.E. and Smith, R.L. (1994), "Simulated annealing for constrained global optimization", *Journal of Global Optimization* **5**, 101-126.

[65] Sasmitra, I. (1995), "Empirical and analytical comparisons of deepest point with stochastic Lipschitz optimization", M.Science thesis, University of Washington.

[66] Seneta, E. (1976), *Regularly varying functions*, Lecture Notes in Mathematics **508**, Springer-Verlag, Berlin.

[67] Shalloway, D. (1992), "Packet annealing: A deterministic method for global optimization. Application to molecular conformation", in *Recent Advances in Global Optimization*, 433-477, Princeton Series in Computer Science, Princeton University Press.

[68] Shen, Z. and Zhu, Y. (1987), "An interval version of Shubert's iterative method for the localization of the global maximum", *Computing* **38**, 275-280.

[69] Shubert, B.O. (1972), "A sequential method seeking the global maximum of a function", *SIAM Journal on Numerical Analysis* **9**, 379-388.

[70] Shepilov, M.A. (1987), "Determination of the roots and of the global extremum of a Lipschitz function", *Cybernetics* **23**, 233-238.

[71] Smith, R.L. (1984), "Efficient Monte Carlo procedure for generating points uniformly distributed over bounded regions", *Operations Research* **32**, 1296-1308.

[72] Solis, F.J. and Wets, R.J-B (1984), "Minimization by random search techniques", *Operations Research* **32**, 1296-1308.

[73] Strigul, O.I. (1985), "Search for a global extremum in a certain subclass of functions with the Lipschitz condition", *Cybernetics* **21**, 193-195.

[74] Stam, A.J. (1977), "Regular variation in $R^d$ and the Abel-Tauber theorem", Reprint, *Mathematisch Instituut,* Rijksuniversiteit, Groningen.

[75] Strongin, R.G. (1973), "On the convergence of an algorithm for finding a global extremum", *Engineering Cybernetics* **11**, 549-555.

[76] Timonov, L.N. (1977), "An algorithm for search of a global extremum", *Engineering Cybernetics* **15**, 38-44.

[77] Törn, A. and Žilinskas, A. (1989), "Global optimization", *Lecture Notes in Computer Science 350*, Springer-Verlag, Berlin.

[78] Wood, G.R. (1991), "Multidimensional bisection applied to global optimisation", *Computers and Mathematics with Applications* **21**, 161-172.

[79] Wood, G.R. (1992), "The bisection method in higher dimensions", *Mathematical Programming* **55**, 317-338.

[80] Wood, G.R. and Zhang, B.P. (1994), "Estimation of the Lipschitz constant of a function", to appear in *Journal of Global Optimization.*

[81] Zabinsky, Z.B. and Smith, R.L. (1992), "Pure Adaptive Search in global optimization", *Mathematical Programming* **53**, 323-338.

[82] Zabinsky, Z.B., Smith, R.L., McDonald, J. F., Romeijn, H. E. and Kaufman, D. E. (1993), "Improving Hit-and-Run for global optimization", *Journal of Global Optimization* **2**, 171-192.

[83] Zabinsky, Z.B., Graesser, D.L., Tuttle, M. E. and Kim, Gun-In (1992), "Global optimization of composite laminates using improving Hit and Run", in *Recent Advances in Global Optimization*, 343-368, Princeton Series in Computer Science, Princeton University Press.

[84] Zanakis, S.H. and Kyparisis, J. (1986), "A review of maximum likelihood estimation methods for the three-parameter Weibull distribution", *Journal of Statistical Computation and Simulation* **25**, 53-57.

[85] Žilinskas, A. (1981), "Two algorithms for one-dimensional multimodal minimization", *Optimization* **12**, 53-63.

[86] Zhang, B.P., Wood, G.R. and Baritompa, W.P. (1993), "Multidimensional Bisection: the performance and the context", *Journal of Global Optimization* **3**, 337-358.

[87] Zhigljavsky, A.A. (1991), "Theory of Global Random Search", Kluwer, Dordrecht.